TECHNICAL OPERATIONS RESEARCH

SOME TECHNIQUES TO HELP IMPROVE METHODS FOR EXERCISING AND EVALUATING COMMAND AND CONTROL SYSTEMS

Peter Kugel and Martin F. Owens

Technical Operations Research
Burlington, Massachusetts

tech ops

DDC AVAILABILITY NOTICES

Qualified requesters may obtain copies from Defense Documentation Center (DDC). Orders will be expedited if placed through the librarian or other person designated to request documents from DDC.

Copies available at Office of Technical Services, Department of Commerce.

DEFENSE DOCUMENTATION CENTER

FOR

SCIENTIFIC AND TECHNICAL INFORMATION

CAMERON STATION. ALEXANDRIA. VIRGINIA

tech/ops

TECHNICAL OPERATIONS RESEARCH

SOME TECHNIQUES TO HELP IMPROVE
METHODS FOR EXERCISING AND EVALUATING
COMMAND AND CONTROL SYSTEMS

Peter Kugel and Martin F. Owens

Technical Operations Research
Burlington, Massachusetts

SUMMARY REPORT
31 January 1964

Prepared for

Directorate of Systems Design
Electronic Systems Division
Air Force Systems Command
L. G. Hanscom Field, United States Air Force
Bedford, Massachusetts

Burlington,   Massachusetts

ABSTRACT

This report describes the development of four techniques intended to improve methods for exercising and evaluating command and control systems:

1. Exercise flow diagram (generalizes and extends the techniques of flow charting for use in exercising)

2. Resource assignment model (uses techniques of mathematical logic to describe the information processing in the command and control system)

3. Expected utility model (uses techniques drawn from decision theory to describe decision making in a command and control system)

4. Finite automaton model (uses techniques from the theory of finite automata to describe the relationships between a command and control system and the system exercising it).

(Appendixes present three of these models in sufficient detail to permit the reader to apply them. The finite automaton model will be discussed in a separate report.) A program is outlined for developing these models during the second part of the contract period.

This Technical Documentary Report has been reviewed and is approved.

D. Parker
1st Lt. , USAF
Project Officer

FOREWORD

The work described in this report (TO-B 63-108) was performed by Technical Operations Research for the Systems Design Laboratory of the Electronics System Division of the United States Air Force under Contract AF19(628)-2455. The purpose of this contract is to develop techniques to improve methods used in constructing, controlling, and evaluating command and control system exercises. The work was based on an examination of records of exercises of an existing command and control system (473L). Models were developed to describe various aspects of these exercises, using existing mathematical techniques. The applications of these models were investigated.

The contract monitor was Lt. Don Parker (ESRC). Martin F. Owens, Robert A. Langevin, Stanley LaVallee, and Peter Kugel (Project Leader) worked on this contract for Technical Operations Research. Department D-25 of The MITRE Corporation, particularly Mr. John Burns and Dr. John Proctor, assisted in making available exercise records and in numerous discussions; this help is gratefully acknowledged. This report describes roughly the results of the first half of the contract. The work described was performed from 15 December 1962 to 30 November 1963.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

THE PROBLEM

The purposes of command and control system exercise and evaluation are:
(1) to give the system experience in carrying out its role, (2) to characterize the
capabilities of the system to permit evaluation by the user, and (3) to provide the
basis for recommendations for improving these capabilities. This report deals with
techniques aimed to help an exercise program fulfill these objectives, particularly
the second and third.

The use of exercises as a basis for design recommendations has become par-
ticularly important as a result of the evolutionary design policy of the Air Force.
According to this policy, large and complex systems are built in increasingly ambi-
tious stages, each stage providing a pilot model for validating concepts to be used
in the design of the next stage.

To obtain data that are useful both for the evaluation of a system as a whole and
for making design recommendations, it is not sufficient merely to watch a system's
performance during an exercise; rather, the exercise must be carefully designed.
The validity of statements about system capabilities and recommendations for sys-
tem improvements depends on the conditions that go into the design of the exercise.
These conditions must be given consideration before the exercise is run.

In one sense, the design of exercises resembles the design of standard equip-
ment tests. We are concerned not so much with tests that determine whether the
manufacturer has met specifications, but rather with those that determine whether
the product meets the needs for which it was procured.

In this report, we are also concerned with the design of command and control
system exercises as scientific experiments. This report describes mathematical
models intended as first steps toward a scientific methodology for the design of exer-
cises viewed as experiments. The use of mathematical models in a scientific theory
has two major benefits:

1. Since the rules for manipulating the symbols that express the model are
precisely defined in terms of the forms of the symbols involved, the results of

manipulating these symbols can be replicated. That is, the results obtained by two different people using the same data are the same.

2. Given any description of a state of the system being modeled, the manipulations of the model describe future states of the system and thus have a predictive power that can be tested in experiments.

Because they have these characteristics, models can provide a framework for evaluating exercises and the systems that exercises are intended to test. A small number of exercises can be used to predict a system's capabilities with respect to the large varieties of problems for which the system was intended[*] and as a basis for determining how system changes affect system performance.

The models described in the appendixes to this report were derived from an examination of the records of some 473L system exercises. Standard mathematical techniques were applied in analyzing the observed behavior of the system. Four models of different aspects of this behavior resulted:

1. Exercise Flow Diagram (Appendix A)

2. Resource Assignment (Appendix B)

3. Expected Utility (Appendix C)

4. Finite Automaton (to be the subject of a separate report).

The value of these models remains hypothetical since they have not yet been applied and tested in actual exercises. The appendixes have been written to be used as manuals by those desiring to try the techniques on particular problems or by those desiring to investigate the techniques further. The appendixes describe the basic notions behind the models rather than the particular version that would be useful for exercising a particular system.

---

[*] Thus the models help provide a "performance envelope" describing the area within which system capabilities lie.

METHOD OF APPROACH

We have adapted the standard scientific method to develop the basis for a technique for command and control system exercising. We began with observation. Since no exercises of the type we wished to observe were planned for 473L during the early part of the contract period, we dealt with the records kept by others who had observed such exercises. From these observations we derived mathematical models that describe the exercises relatively precisely. These mathematical models of specific exercises were then generalized in an effort to provide a framework within which many types of exercises might be studied.

The main purpose of a model is to predict the behavior of that part of the world it is simulating. A model often is intended, not merely to predict this behavior, but also to predict how changes in certain elements in the environment will change the behavior of the system. It can do this because it is a simplified version of the real world and its parts are clearly and precisely defined. The model is expressed in symbolism that is easily manipulated using the machinery of modern mathematics.

The models we have developed are similar to traditional mathematical functions. However, they differ in a number of details. These differences are important because they involve using mathematical machinery that may not be familiar to some readers of this report. The differences include the following:

1. The ranges and domains of these models tend not to be continuous as are the ranges and domains of the standard functions of analysis.

2. The ranges and domains of these models are seldom linearly ordered in a natural way as the real numbers are, and there is seldom a natural metric on the range and domain.

3. The ranges and domains of these models fall into a large number of quite different and not comparable classes, whereas the ranges and domains of the more familiar functions are usually taken from a single class (e.g., the real numbers).

Because of these three differences, one cannot assume all the structure of real numbers, and thus one cannot use all the machinery built up to manipulate and study real numbers in traditional mathematics. Nevertheless, one can borrow some of

those aspects of mathematical systems that make them useful in scientific theories. The principles we will utilize include the following:

1. The structure of the model itself is to be made precise and explicit so that it can be matched against the known elements of any actual situation to determine whether the axioms upon which the model is based actually apply. By insisting on this, one can know exactly where one's theory will be applicable.

2. The model not only is a simplification of the real situation, but it is also described in terms of symbols that can be manipulated easily on blackboard or paper (or on digital computers, for very large problems). The model can thus be studied in ways in which the real system cannot.

3. Although one may have ranges and domains not as well structured as the real numbers, one may find somewhat weaker mathematical orderings in them. Thus one can use some existing mathematical results.

Once the utility of models as tools for improving exercises for command and control systems is accepted, two basic decisions must be made: (1) What independent variables of a command and control system is one going to relate to what dependent (behavioral) variables? (2) What kind of mathematical machinery will one use in order to express the model itself? In the following sections, we will discuss briefly our reasons for developing the specific models presented in this report.

FLOW CHART OF EXERCISE (FCE) TECHNIQUE

We began by flow-charting the exercises we were using as raw materials, first simply to organize the observations on which models were to be based. In other words, flow-charting was used as a method for keeping notes. It occurred to us that the repertoire of shapes and connectors available in the usual flow-charting techniques was too limited if we wished to represent much of the logic of an exercise by means of the shapes and the interrelationships of the boxes alone, rather than in terms of what one wrote into those boxes. Since computer flow charts are used to represent a small fragment of mathematical logic, it was a natural thought to extend this fragment. As a result, we began to introduce some logical functions explicitly into the machinery of the flow chart (those corresponding to the logical "and," "or," "if then," and so forth). We also began to distinguish between various types of

operations in ways not necessary when making flow charts of computer programs. Thus, we borrowed the usual distinction between facts (data) and system actions (operations). However, we added to these the additional notion of a motivating factor, which a computer program generally does not need (but which a system run by human beings does need).

This technique, which we shall call the "flow chart of exercise (FCE) technique," was designed to show several facets of an exercise:

1. System inputs and outputs (messages in 473L)

2. System actions

3. Activators or motivators which caused (2) in response to inputs (1)

4. Means of combining (1) and (3) to cause (2).

One 473L exercise[*] was diagrammed using the FCE[†] technique. Table 1 is a key to FCE symbols, and Figure 1 is a sample chart using this technique. These are included for the benefit of the reader who may prefer this technique to the EFD technique into which it later evolved.

EXERCISE FLOW DIAGRAM (EFD) TECHNIQUE

It occurred to us later that the FCE technique could be used in design, control, and evaluation, and that the distinctions we had made for the purposes of recording our observations were not as basic in this second role as we first thought they might be. We discovered that the distinction between what is inside a system, and what is not, is not so basic as the distinction between the roles that various items of information play. Thus, we amalgamated those facts presented to the system about the world before the exercise with those presented during the exercise, since they both describe the external state of affairs. We also changed the structure so that it would

---

[*] BANKO I

[†] The FCE technique was described in our Concept Papers 1 and 2. Although familiarity with the concept papers written during the contract is not presupposed in this report, we will indicate where the contents of this report overlap the contents of those papers.

TABLE 1
KEY TO FCE SYMBOLS

DATUM

ACTION

DECISION

ACTIVATOR

CONNECTOR, ALL INPUTS REQUIRED

CONNECTOR, ONLY ONE INPUT REQUIRED

CONNECTOR, INFERENCE

DEAD END

B   MESSAGE DESIGNATION

P   SPECIAL CASE, PROBLEM IF THIS PART ENTERED

ONE-WAY FLOW OF LOGIC OR INFORMATION (LEFT TO RIGHT)

TWO-WAY INFORMATION FLOW (UP, THEN DOWN)

TWO-WAY INFORMATION FLOW (DOWN, THEN UP)
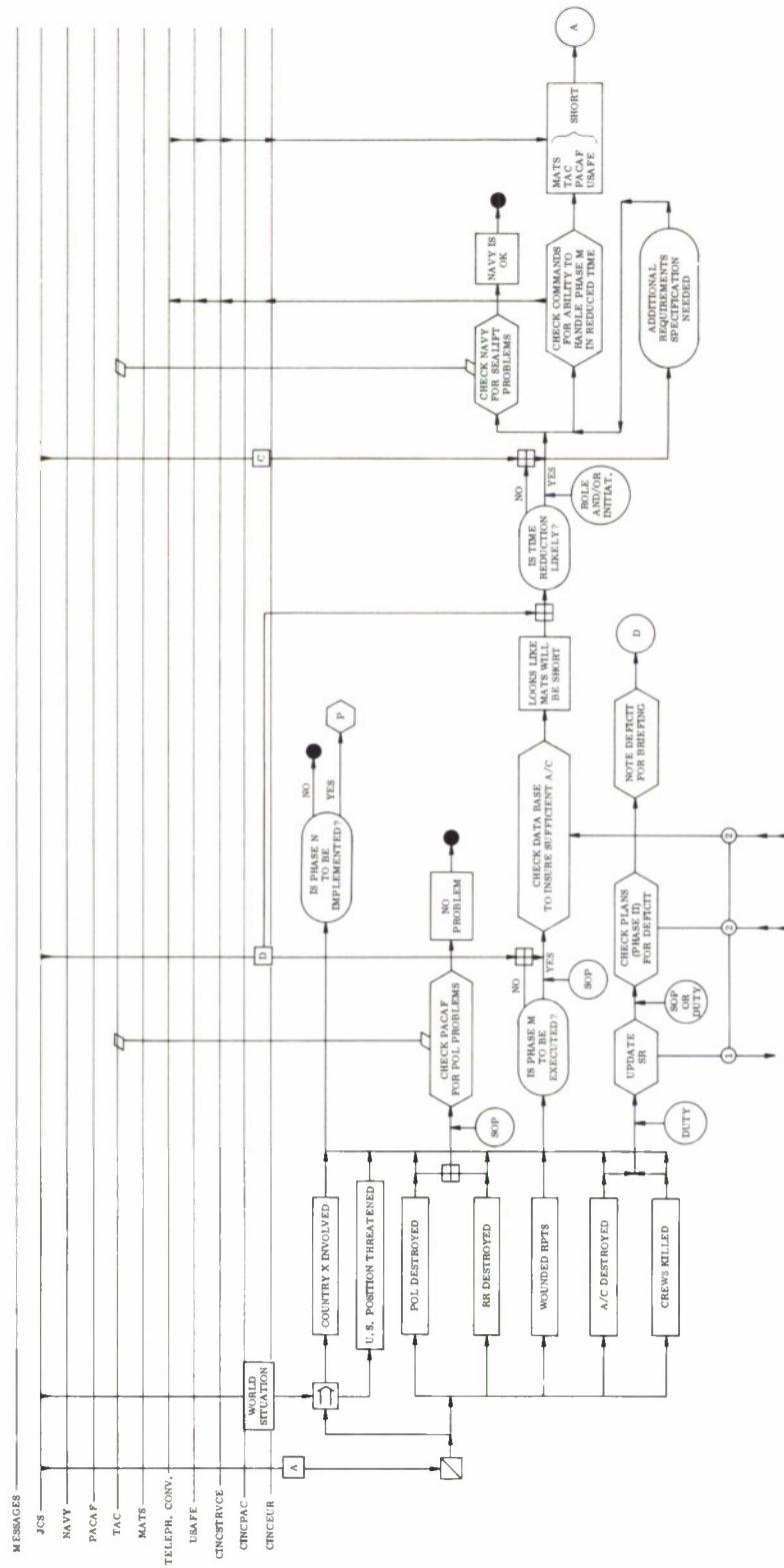
Figure 1.   Sample FCE Chart

have greater utility for use in designing exercises whose results would place a system on some sort of a scale, rather than merely on one side of a succeed/fail boundary.

The exercise flow diagram technique also appeared to have uses beyond those of exercise analysis (for which it had first been used) and exercise design (for which we later found it might be useful). Various features were added to facilitate these applications. For details see Appendix A.

RESOURCE ASSIGNMENT MODEL

The resource assignment model arose from the observation that a command and control system is basically an information processing system. Looking at 473L, we noted that its inputs were information about resources that tended to define problems, and that the outputs of the system were either the solution of these problems or the information upon which rational decisions leading to the solutions of these problems could be based.

In the last 5 years, considerable work has been done in the general study of problem solving. In particular, Newell, Shaw, and Simon[1] have developed a general problem solving model (GPS). We took their model and tried to specify it in such a way that it could describe the solving of resource assignment problems. Observing that much of what went on in the 473L exercise consisted of transformations of objects in a four-dimensional space-time continuum, we considered using vector notation to represent these transformations. We tried to use additional dimensions on the vectors to represent other variables (missions, resource characteristics) important in the specific exercise examined most closely. However, at this point we began to realize that the utility of this notation depends on the similarity of the dimensions being represented to the continuum of the real numbers. Since this was not the case with resource characteristics, the machinery soon became unwieldy. Next we turned to the techniques of modern mathematical logic.

We began using the machinery of the first-order predicate calculus. However, this proved too weak since we were often concerned with properties of properties. We knew that importing all the machinery that might be useful to discuss the properties of properties was a dangerous procedure, tending to lead to paradox. As a result, we took the intermediate course of using a higher order predicate calculus with a theory of types plus typical ambiguity as described in Appendix B.

EXPECTED UTILITY MODEL

The expected utility model arose because we thought that considering command and control systems as decision-making systems might be a good way to relate certain of their properties. We considered the usual mode of rational behavior derived from Von Neumann and Morgenstern.[2] The results of this application are described in Appendix C. However, a major difficulty in the application of such a model is the difficulty of assigning or determining in any way whatsoever, either the numerical probabilities of alternatives or the numerical value of the utilities.

FINITE AUTOMATON MODEL

The finite automaton model, which will be presented in a separate report, originated in a manner somewhat different from that of the other models. This model began with the formalism (in this case the notion of coupled automata). It was developed by Robert A. Langevin working under the Corporate Fellowship program of Technical Operations Research, largely independently of contract AF 19(628)-2455, although he was guided partly by information derived under that contract. The result of this approach was a particularly elegant and content-free model which has, perhaps, more mathematical interest than the other models developed, but which may at the same time have fewer practical applications.

In Chapter 2, we shall describe each of these models in more detail and discuss their applications. Recommendations for future work will be given in Chapter 3.

CHAPTER 2

FOUR MODELS AND THEIR APPLICATIONS

INTRODUCTION

Each of the four models developed under this contract describes relationships between certain characteristics of command and control systems. These models are like mathematical functions that relate controllable characteristics (independent variables) to other characteristics (dependent variables). In general, the independent variables will be the characteristics of a problem presented to a system and the dependent variables will be the responses of the system to that problem. The four models differ both with respect to the characteristics they relate and, less importantly, to the mathematical machinery they employ.

USES OF MODELS

These models predict system responses to many problems, based on the observation of system responses to only a few problems. These predictions define a kind of "boundary" within which system capabilities are believed to lie. Such a boundary is a system characterization that is more than a simple determination of whether a system is satisfactory, but less than a precise numerical measure of system capabilities. Its purpose is to provide system users with a basis for system evaluation.

Predictions of system behavior may also help anticipate control problems that may occur during the running of a planned exercise and the kinds of behavior that monitors might look for during exercises. By making the elements involved explicit, these models may provide the basis for a more conscious and perhaps better-reasoned choice of exercise parameters. By providing a language to discuss the elements of exercise design, these models may also supply a basis for dividing the job of exercising among various people and a medium for better communication between the exerciser and the user during pre- and post-exercise briefing.

In addition, such models, by categorizing the elements of exercises and making them precise, may furnish a vehicle for defining algorithms. These algorithms may be used to automate certain aspects of exercise construction. They may also be used to organize the data base in exercise control and may provide a vehicle for the

derivation of precise mathematical results defining certain absolute limits[*] beyond which systems cannot conceivably go.

Finally, these models may have uses in areas other than command and control system exercising. One application which has already been mentioned is command and control system design. Some thought has been given to the possibility of using the finite automaton model to predict the results of binary choice experiments, and using the resource assignment model to study certain aspects in the general theory of problem solving.

CHARACTERISTICS

Table 2 lists the main characteristics of the four models developed under this contract. (These are summarized in the remainder of this chapter and the first three are described in detail in the appendixes.) The table is not complete, since it contains only one entry in each block. It is intended to help the reader choose the technique best serving his need.

TABLE 2
FEATURES OF THE FOUR MODELS

Appendix	Model	Describes	Relates	Main Application	Main Mathematical Discipline
A	EFD	Relationship between scenario, mission, and system actions	Exercise plan to system response	Conceptual tool for exercise design	Boolean algebra
B	RA	Information processing in command and control system	Problem statement to possible solutions	Evaluation of exercise results with respect to information processing	Mathematical logic
C	EU	Decision making in a command and control system	Information plus system goals to system decisions	Evaluation of exercise results with respect to decision making	Probability theory
*	FA	Control of system during exercise	Amount of control to effectiveness of control	Evaluation of exercise as a learning experience	Theory of finite automata

[*] To be the subject of a separate report.

---

[*] Similar to some results in the theory of uncomputability.

THE EXERCISE FLOW DIAGRAM TECHNIQUE

DESCRIPTION

The exercise flow diagram (EFD) technique is an extension of the notion of flow charting used in computer programming. An exercise flow diagram consists of a series of elements (the boxes in traditional flow charts) and relationships between these elements (the directed lines in flow charts) for graphically expressing the basic logic of an exercise. By "logic of an exercise" we mean the general principles governing the flow of information in an exercise. As in computer flow-charting, certain aspects of this logic are made explicit by means of the shapes and directions of the items of which the chart is constructed. The items are drawn from a finite vocabulary of shapes and thus provide an easily understood, but formal, language. The flow diagrams based on this technique depict separately the logic of the scenario, the logic of system motivations (the system's sense of mission), and the logic of system actions. In general, a system action occurs if, and only if, there is an appropriate combination between an element in the scenario and a part of the system's mission.

TECHNIQUES USED (EFD)

The logic depicted in an exercise flow diagram is basically that of the first-order predicate calculus, much as the logic of computer-programming flow charts might be said to be the logic of truth functions.

APPLICATIONS (EFD)

The most immediate benefit to be gained from the EFD technique is the ability to help organize and categorize the elements involved in system exercise design. This technique helps to provide a language for explicitly representing certain relationships between exercise elements at an early stage of exercise design. This explicit representation may then provide a medium for the more rational assignment of values to various parameters of exercise design. As stated previously, the EFD technique may also help provide a better medium for communication between the various people involved in actual construction of exercises.

The EFD technique can also be used as a general technique for designing exercises as experiments to determine the performance characteristics of large command

and control systems. Its use as a tool for exercise design may vary, depending on the philosophy of system exercising. In the spirit of normative[3] exercising, the technique provides a method for the orderly construction of exercises. One can proceed initially with the logic of the scenario (the problem), which constitutes one segment of the chart. This can then be related to system performance, or at least to anticipated system performance, by means of messages whose roles can be described in the formalism. The contents of these messages can be compared with elements of the system's mission, in consultation with the user to determine the adequacy of these contents. The check points can be indicated, and the contingency messages ordered in terms of increasing force. Such a technique also provides a means for checking the adequacy of the series of contingency messages developed prior to running the exercise. In addition, the technique may provide a method for the orderly construction of exercises to produce a performance "envelope."

When the logic of the elements of an exercise have been made explicit, an exercise flow diagram may become a useful tool for exercise monitoring. The diagram shows how the various elements of an exercise fit together and what purposes they serve. It is a convenient way of keeping track of the corrections to be made if the system wanders off the intended path, and also serves as a worksheet for recording observed system behavior during the running of an exercise. This particular form of record might be useful during briefings because it provides a graphic tool capable of displaying all the elements of the exercise. Finally, an exercise flow chart might be translated into a more mathematically tractable notation (predicate calculus, for example) and used as the input for a computer program to automate certain parts of exercise construction or control.

THE RESOURCE ASSIGNMENT MODEL

DESCRIPTION

The resource assignment (RA) model describes the information processing that occurs in the system during the course of an exercise. The notation is considerably less graphic (and therefore appears more formidable) than that of the exercise flow diagram model, but it is considerably more powerful. (For example, the model might provide a reasonable vehicle for a general problem-solving program, on the order of the general problem solver of Newell, Shaw, and Simon,[1] which it closely

resembles.) In this model, the activity performed by a command and control system is characterized as a kind of problem solving, and an effort is made to make each step involved in this process explicit. The model consists of a description of the initial state (which describes the state of the world when the problem is set), a final state (which describes the state of the world when the problem has been solved), and a series of allowable transformations that one applies to change the first into the second. A solution of a problem is seen as a series of instructions for applying the transformations to the initial state to yield an appropriate final state.

TECHNIQUES USED (RA)

The reader will perhaps recognize the similarity of the description of a problem given above to the modern characterization of a formal (axiomatic) system. The initial states correspond in this analogy to axioms, transformation rules to rules of inference, and the final states to theorems. The series of instructions for applying transformations to yield a solution correspond to proofs.

The devices used in the study of such systems (the techniques of mathematical logic) are also used to describe this model, although certain notational changes are made to simplify matters. The machinery used is a fragment of a higher order predicate calculus that is sufficiently weak to avoid paradox.

APPLICATIONS (RA)

The RA model defines the relationships between a problem and its solutions. Such a definition may be useful in the following applications:

1. Exercise design, to help:

    a. Determine whether the information to be given to a system during an exercise is sufficient to put the intended problem to the system.

    b. Define the values of those parameters of the problem it may be desirable to control during the design phase, such as the difficulty of recognizing that a problem exists or the difficulty of solving a problem.

    c. Determine whether an exercise plan really controls system behavior in the manner desired.

2. <u>Exercise monitoring and control</u>, to help:

    a. Organize control or contingency messages by classifying them according to their roles. This classification may be useful in planning manual or automatic exercise control.

    b. Organize monitoring by predicting the logically possible system behavior so that this behavior can be anticipated.

3. <u>Evaluation of exercise results</u>, to:

    a. Provide an ideal for comparison with observed performance.

    b. Help organize a description of these results for use in making recommendations.

4. <u>System evaluation and recommendations</u> for system improvement, to provide:

    a. A medium for the reallocation of system functions on the basis of exercise results, by providing a description of system functions relatively independent of implementation.

    b. A basis for system evaluation by a user by generalizing exercise results to a large class of problems, given only the observations of system behavior in response to several problems.

5. <u>Design of command and control systems</u>, to provide:

    a. A medium for allocating functions among the various elements of a system.

    b. The basis for the organization of more flexible command and control systems in which the user defines his problem when it arises.

THE EXPECTED UTILITY MODEL

DESCRIPTION

The expected utility (EU) model is based on the utility theory developed by Von Neumann and Morgenstern.[2] It describes the elements that go into the process of rational decision-making under conditions of incomplete or unreliable information. A decision is seen as the choice of one of several alternate actions, where the consequences of each of these actions are not completely known and where the values of the consequences may not be completely known. Decisions are defined as functions of the system characteristics, and of the probabilities and utilities of their outcomes.

TECHNIQUES USED (EU)

The machinery applicable here is probability theory, limited because of the difficulty of establishing reliable numerical values for the probabilities and utilities involved.

APPLICATIONS (EU)

The EU model appears to be useful mainly in the exercise and evaluation of the decision-making function of a command and control system. Since decisions are usually made by key people, this model may be appropriate for determining and evaluating the characteristics of key personnel in a command and control system. Because electronic data processing provides the basis for the decisions of these people, the EU model may also provide a method for evaluating the outputs of the computers.

The input to the EU model is a precise description of the particular problem to be presented during an exercise. This description provides another conceptual framework for use in exercise design. Given this description, the model makes a hypothesis of what constitutes ideal behavior for the exercise, and this ideal is a standard against which actual performance can be measured. Once this has been done, the actual performance of the system can be described by changing certain aspects of the version of the model used to predict ideal behavior. The model thus changed is a predictor of system behavior (response) for a large class of problems (stimuli). This is a performance envelope that can provide not only the basis for a user evaluation of the system (will it behave as he would like it to?) but also the

basis for recommendations for system changes to produce improved behavior. One such recommendation can be further exercising. The model may provide a measure of the efficacy of such exercising as a learning experience.

<div align="center">THE FINITE AUTOMATON MODEL[*]</div>

DESCRIPTION

The finite automaton (FA) model is based on the theory of finite automata. This model is perhaps the most abstract or content-free model of the four presented in this report. It describes the relationships between the systems being controlled during an exercise and the system attempting to control the exercise. It relates the amount of information obtained by the controlling system, and the amount of information this system gives to the systems being controlled, to the effectiveness of control. It also relates various parameters of the exercise to the efficacy of the exercise as a learning experience for the command and control system. It may be used to study the effectiveness of different command and control system strategies.

TECHNIQUES USED (FA)

The machinery used is that of the theory of finite automata. The model considers the effects of coupling such automata, where coupling occurs only to transfer information from one automaton to another. Each element of the model — the problem, the command and control system, the command and control system's image of the world, the exercise controller, the exercise controller's image of the problem — are expressed as finite automata. The model is determined by the strategies used in its various elements. However, because the user may choose the strategy to be used (although once he has chosen it, it is completely defined), this model provides him with a medium for investigating the values of alternate strategies by comparing the model's behaviors as it operates under various sets of rules.

APPLICATIONS (FA)

The FA model is perhaps best used for the study of relationships between rather certain general choices of parameters and the resultant behavior of the command and

---

[*] The details of this model will be described in a separate report.

control system (both during an exercise and during actual operations). This model may be useful in anticipating problems in control that might arise during an exercise, given the exercise design. It may help in planning exercises that will be more satisfactory learning experiences, and may be useful in the study of overall command and control system strategies. The FA model is particularly well suited to computer simulation.

CHAPTER 3

RECOMMENDATIONS

The four models developed during this contract are tools that may become the basis for a precise and scientifically sound methodology for use in almost all phases of command and control system exercising. However, the work described in this report is incomplete and requires development in at least four directions:

1. The models themselves need to be more fully developed, and particular models need to be written for particular control systems.

2. The actual utility of the models for exercising has to be investigated. We hope that this can be done by having those who construct exercises use the appendixes of this report in their work.

3. Certain characteristics of the models (as contrasted to characteristics of the systems the models describe) need to be investigated more fully. For example, it seems feasible to count certain types of boxes in an exercise-flow-diagram representation of an exercise as a measure of the complexity of that exercise.

4. The hypothesis that these models can provide bases for predicting command and control system behavior needs to be tested by experiment.

The preceding items suggest the following continued work:

1. Develop models for specific systems.

2. Apply these specific models to determine their utility to their users.

3. Investigate and develop model characteristics, and use these characteristics to define intuitive concepts more precisely.

4. Validate models as predictors of system behavior.

These four items will be the basis for work to be carried out under the remainder of this contract.

REFERENCES

1. A. Newell, J. C. Shaw, and H. A. Simon, "Report on a General Problem-Solving Program," Proc. Internatl. Conf. on Information Processing (Paris, France: UNESCO House, 1959).

2. John Von Neumann and Oskar Morgenstern, "Theory of Games and Economic Behavior" (Princeton, New Jersey: Princeton University Press, 1953).

3. J. H. Proctor, "Normative Exercising: An Analytical and Evaluative Aid in System Design," paper delivered at the First Congress on the Information System Sciences, Session XV, Hot Springs, Virginia (20 November 1962).

APPENDIX A

THE EXERCISE FLOW DIAGRAM (EFD) TECHNIQUE

APPENDIX A

THE EXERCISE FLOW DIAGRAM (EFD) TECHNIQUE

INTRODUCTION

The exercise flow diagram (EFD) technique is a method for producing stand-
ardized, graphic descriptions of the structures of exercises of command and control
systems. This technique, which resembles the flow-charting technique used in com-
puter programming, may be useful in the design and construction of exercises. It
may be helpful as a graphic debriefing tool for postexercise evaluation, and could
provide the basis for more powerful and flexible techniques for making design recom-
mendations based on exercise results.

The EFD technique is intended to simplify the job of producing exercises. It
provides a common language of communication for use by those designing and con-
structing an exercise. It may provide a basis for checking the adequacy of proposed
exercises relative to a system's mission and capabilities. It may be useful in help-
ing to control exercises by providing a convenient way of keeping track of the prog-
ress of an exercise and of the pre-planned measures to be taken to control this prog-
ress. These functions of the EFD methodology will become clearer once the method
itself has been explained. The end of this appendix will explain how these intended
benefits might be obtained.

The technique is based on the notion of a formalized flow chart of system exer-
cises.[*] A "formalized flow chart" consists of a set of different shapes that have
been taken from a fixed "vocabulary" of shapes defined beforehand. These shapes
and their meanings are summarized in Table A-1, p. A-2. The shapes are con-
nected by flow lines, and rules may be provided to govern the ways in which shapes
may be connected to each other.

Formalized flow charts are intended to describe the <u>relationships</u> between
operations, information, messages, plans, rules, and similar items involved in the
solution of an exercise problem. At this stage, the intent is not to formalize the

_____

[*]The rest of this Introduction is adapted from our Concept Paper No. 1.

TABLE A-1

SUMMARY OF EFD NOTATIONAL DEVICES

THE AREAS OF THE CHART

①   Exercise Problem (World):   This is represented in the top part of the chart and contains states of affairs and their logic.

⑴ₐ   a. Messages: These are represented by vertical lines between sections ① and ② . Their characteristics are indicated in the space between ① and ② .

②   System Actions (Behavior):   These and their logic are represented in the middle of the chart.

③   System Mission (Activators):   These are represented on the bottom of the chart.

ELEMENTS OF THE CHART

The meanings of some shapes depend on their locations in the chart (whether they are in parts ① , ⑴ₐ , ② , or ③).

Symbol	Meaning of Symbol
	In Part① : A state of affairs
	In Part⑴ₐ : A description of a state of affairs (contents of messages)
	In Part③ : An activator

Symbol	Meaning of Symbol
	A is a specific example (specification) of B
	A is implied by B, with probability P
	A occurs if, and only if, B occurs
	System Action (used only in ②)

SUMMARY OF EFD NOTATIONAL DEVICES

In the Message Area:

●   All of the message is devoted to the fact.

◑   Half of the message is devoted to the fact.

◕   One quarter of the message is devoted to the fact.

⊖   or   ⊍   Implies

[A]—⊖—[B]   A implies B

⊕   And
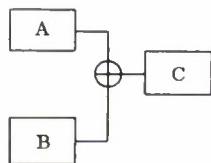
A and B are required for C

⊙   Or
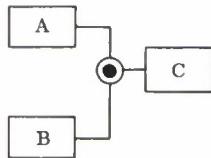
Either A or B are required for C

In the Middle of the Chart

○   Observable behavior

(Possibly observable behavior

nature of the operations, information, messages, plans, and rules themselves. (This is the subject of Appendixes B and C.) In other words, the purpose of the EFD technique is to describe how the elements of an exercise fit together, rather than what they actually are.

This formalization is intended to be such that a person who does not understand the particular exercise diagrammed should be able, by looking at the rules of the formalization and the diagram, to understand the aspects of an exercise expressed in the chart. Producing a formalized flow chart is similar to the way in which one formalizes a problem before applying a particular branch of mathematics to it. For example, a high-school algebra problem formalized by

"Assume A + B = C. Given A and C, find B."

does not require that a person know what A, B, and C represent in order to apply the albegra required to solve the problem. It is quite different to derive the formal version from the informally stated problem. This appendix deals primarily with this task,which requires the user's intuition and, therefore, cannot be taught in the same way as the manipulations of algebraic symbols.

BASIC CONCEPTS

A system action is viewed as a result of combining a fact (or a "state of affairs") and a rule (or an "activator"). An exercise is based on a scenario, which is a series of states of affairs whose occurrence is simulated during the exercise. This series of states (the events in the external world) produces a series of consequences, some of which are transmitted to the system. These consequences are called messages. By receiving these messages, the system gets an image of the state of affairs, which may or may not be accurate. States of affairs (SOA) are given by sentences such as "It is raining in Laos"; "It will rain in Okinawa on Tuesday"; "Plan X says do so and so"; "General Y thinks that Z is a good thing." States of affairs are facts; they do not involve value judgments or imperatives. "The system has been ordered to do X" is a state of affairs; "The system must do X" is an imperative. Similarly, "The President thinks that Y is desirable" asserts a state of affairs; the statement "Y is desirable" is a value judgment. This distinction is important for an understanding of the EFD technique.

States of affairs may imply other states of affairs. Thus, the SOA "It is raining in Laos" together with the SOA "Vientiane is in Laos" suggests that it is possible that the airfield at Vientiane is closed. A group of SOA's and the relationships between them are the skeleton of a scenario. These are represented in the top part of exercise flow charts.

A system has an image of what it is supposed to do. This image can be expressed as a collection of general rules such as "Obey any order from a superior officer" or "Monitor critical situations." All these rules can be translated into statements of the form "If a state of affairs of the type XXX happens, then respond to it by doing YYY." Such rules we call <u>activators</u>. The steps leading to a system action are as follows:

1. As a result of receiving a message, the system discovers that a state of affairs X has occurred.

2. It reasons that this event is of type XXX.

3. It combines the result of Item 2 with the activator and produces a system action.

Thus, in the case of an activator of the form "Obey any orders from a superior officer," a message of the form "General Z orders the system to do YYY" leads to the action YYY only if the system (a) believes General Z to be a superior in the chain of command and (b) believes that the message really comes from General Z.

An exercise consists of presenting a scenario to the system and observing the system's response to this presentation. As a result of such a presentation-plus-observation, one hopes to be able to determine system characteristics in various directions and to compare these with requirements. This process is called an <u>evaluation</u> of the system.

THREE AREAS OF AN EXERCISE FLOW DIAGRAM

An exercise flow diagram is divided into three areas by two horizontal lines as shown in Figure A-1. The upper part of an EFD represents the states of affairs and their logic as they are to occur with the passage of time during an exercise; the bottom part represents aspects of the system's mission (the activators and their logic).

The middle part represents system actions, a result of combining the elements from the top and the bottom. Vertical lines from the top to the middle represent system inputs (messages), which apprise the system of the states of affairs represented in the top part of the chart, and outputs by means of which the system affects the (simulated) world.

① EXERCISE PROBLEM (WORLD)

①a INPUTS (MESSAGES) AND OUTPUTS

② SYSTEM ACTIONS (BEHAVIOR)

③ SYSTEM MISSION (ACTIVATORS)
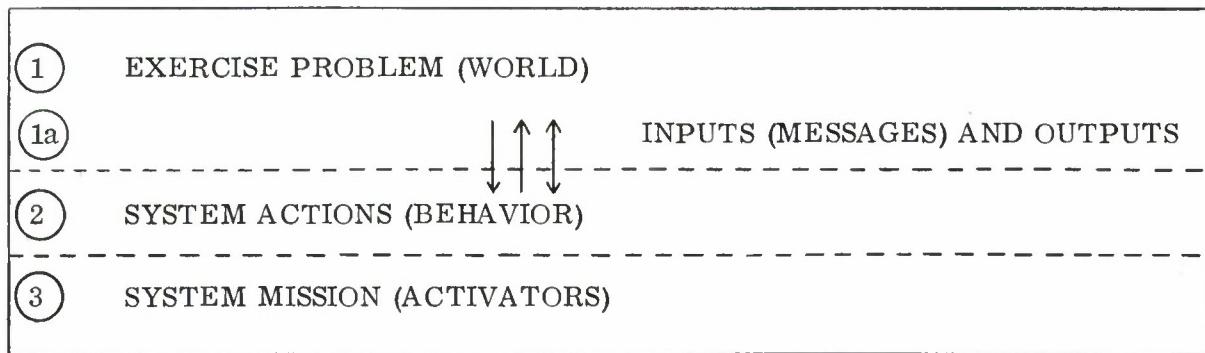
Figure A-1. The Three Sections of an Exercise Flow Diagram

The interrelationships between the three parts of the chart may perhaps best be explained by an example. Let us assume that we are designing an exercise in which the system's problems will be based on meeting the requirement for an airlift into West Berlin. Assume that we have designed a scenario which begins with the stopping of a truck convoy into West Berlin. Figure A-2 shows this latter state of affairs represented by a box in the top part of the flow diagram. As the exercise progresses, this state of affairs will become more and more apparent to the system as it receives more and more messages. The first message may be the report of a rumor; the second may be a statement that something is going on at a specific location, but containing no details; the third may be à specific report.

The fact that a particular convoy has been stopped may be a part of a larger state of affairs, such as a blockade of West Berlin.* This larger state of affairs may lead to another state of affairs (for example, an airlift), in which the command

_____

*The manner in which this is represented is discussed on p. A-9.

and control system being exercised will play a part. These relationships between SOA's are a part of what might be called the logic of the scenario. An EFD would represent this logic as shown in Figure A-3.

(1) Convoy X Stopped at Point Y

(1a)- Message Reporting Rumors of Stoppage Somewhere — Overheard Message that Something Seems to be Going on at Point Y — Message Stating that Convoy X Has Been Stopped at Y --

(2)
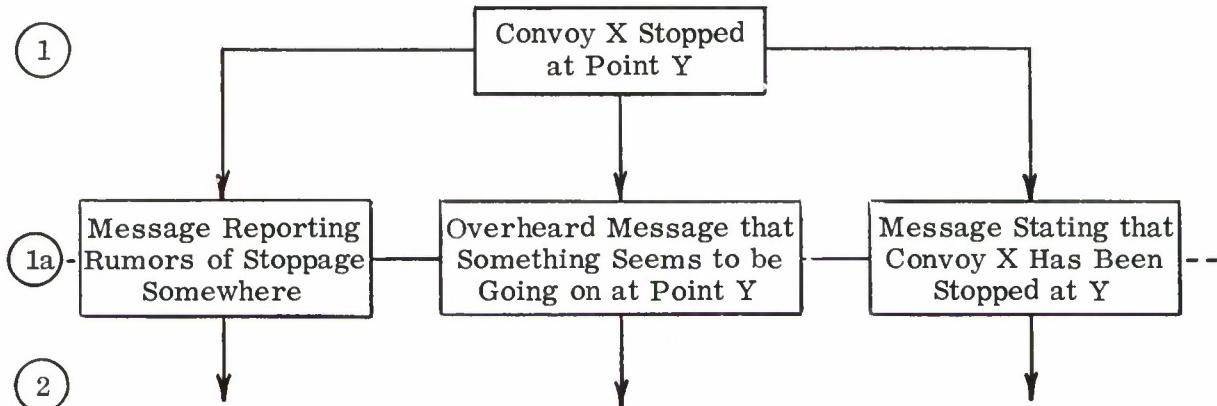
Figure A-2. Messages Conveying a State of Affairs

(1)   BLOCKADE   --- likely --->   AIRLIFT

(2)
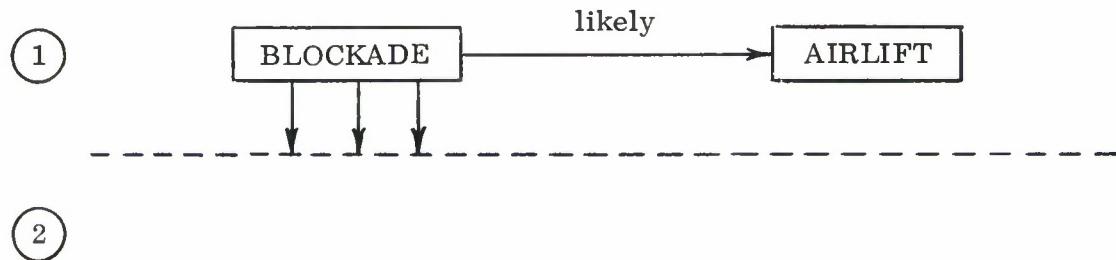
Figure A-3. An Aspect of the Logic of a Scenario

Events follow one another in a scenario because that is the way the scenario has been written. It has been written that way because there is a certain logic in such a series of events. Thus an airlift might follow a blockade because this is a way that the U.S. might respond to such a situation. However, it is not certain that, given a blockade, the U.S. response would be an airlift. The system's ability to anticipate events in the scenario depends in part on how likely these are in the light of events that have already occurred. In some cases it may be useful to write the degree of this likelihood on the directed line indicating the connection, as in Figure A-3.

Assume that one expects the system to alert Officer Z once it realizes that there is a blockade of West Berlin. Such an action of the system requires not only the awareness of the state of affairs ("There is a blockade in West Berlin") but also an awareness of an aspect of the system's mission (an activator), such as: "When there is a blockade in West Berlin, alert officer Z." Activators are represented in the bottom third of an EFD.

The structure of the lower part of the diagram, plus the system's actions in response to the scenario, indicates the system's alertness or awareness of its mission. If the system does not call Officer Z until a state of affairs "General Y orders the system to call Officer Z" occurs and combines with the activator "Obey orders from superior officers," the system is less sensitive (less alert) than if the system responds to the state of affairs "Two truck convoys into West Berlin have been stopped" (which may combine with the activator "Call Officer Z if a serious crisis appears to be in the offing."). It is more difficult to realize that a state of affairs in Berlin may lead to Air Force action than to recognize an order. Given this general mission statement and given the messages indicating the blockade, a system action may be expected here if the system is "thinking ahead" (Version 2 of Figure A-4).
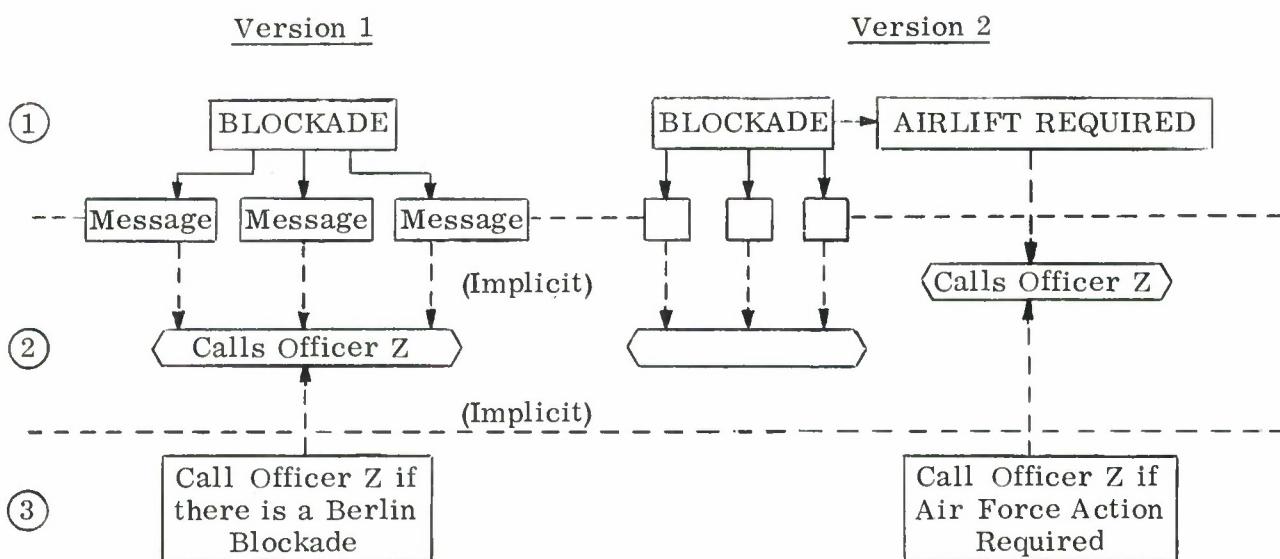
Figure A-4. The Elements Required for System Action
(The dotted lines marked "Implicit" are not drawn in an actual diagram.)

One function of an exercise is to determine precisely at what point the system does act. Some system actions may not be observable. From the observable actions, certain system behavior will be selected by exercise planners for observation during the course of the exercise. The actions so selected are called "check-points"[1] and are marked in the central section of the diagram. The observation of when these check-point actions occur, together with the exercise flow diagram which describes the elements that lead up to this check point, constitutes one of the major inputs to the evaluation of a system.

ELEMENTS OF THE DIAGRAM

This section describes the shapes of boxes and kinds of lines used in an exercise flow diagram. Many of these items are used in all three parts of the diagram; others are peculiar to individual sections. These elements are summarized in Table A-1, p. A-2, and are illustrated in the sample diagram of Figure A-5, p. A-10.

A _rectangular box_ represents a state of affairs. Examples are: "An airlift occurs," "A blockade occurs," or "The U.S. position is threatened militarily."

A _box inside another box_ represents what we shall call "specification." This indicates that the state of affairs represented by the interior box is a more special case of the state of affairs described in the larger box. For example, the fact that "A certain truck convoy has been stopped at a certain check point" is a special case of the fact that "All road traffic has been stopped into West Berlin." The fact that "A different truck convoy has been stopped at a different point" is also an example of the general state of affairs. However, the second stoppage is not an example of the first stoppage; therefore, these two elements both would appear inside the general "Traffic Stopped" block, as illustrated in Figure A-6. In general, it will not be clear whether the smaller fact conveys more information to a system than the larger one or not. Both possibilities exist. In Figure A-5 (p. A-10) the general state of affairs is that "The U.S. position is threatened militarily." This communicates very little

---

[1]This is a slight deviation from the terminology of Proctor. See J. H. Proctor, "Normative Exercising: An Analytical and Evaluative Aid in System Design," paper delivered at the First Congress on the Information System Sciences, Session XV, Hot Springs, Virginia (20 November 1962).
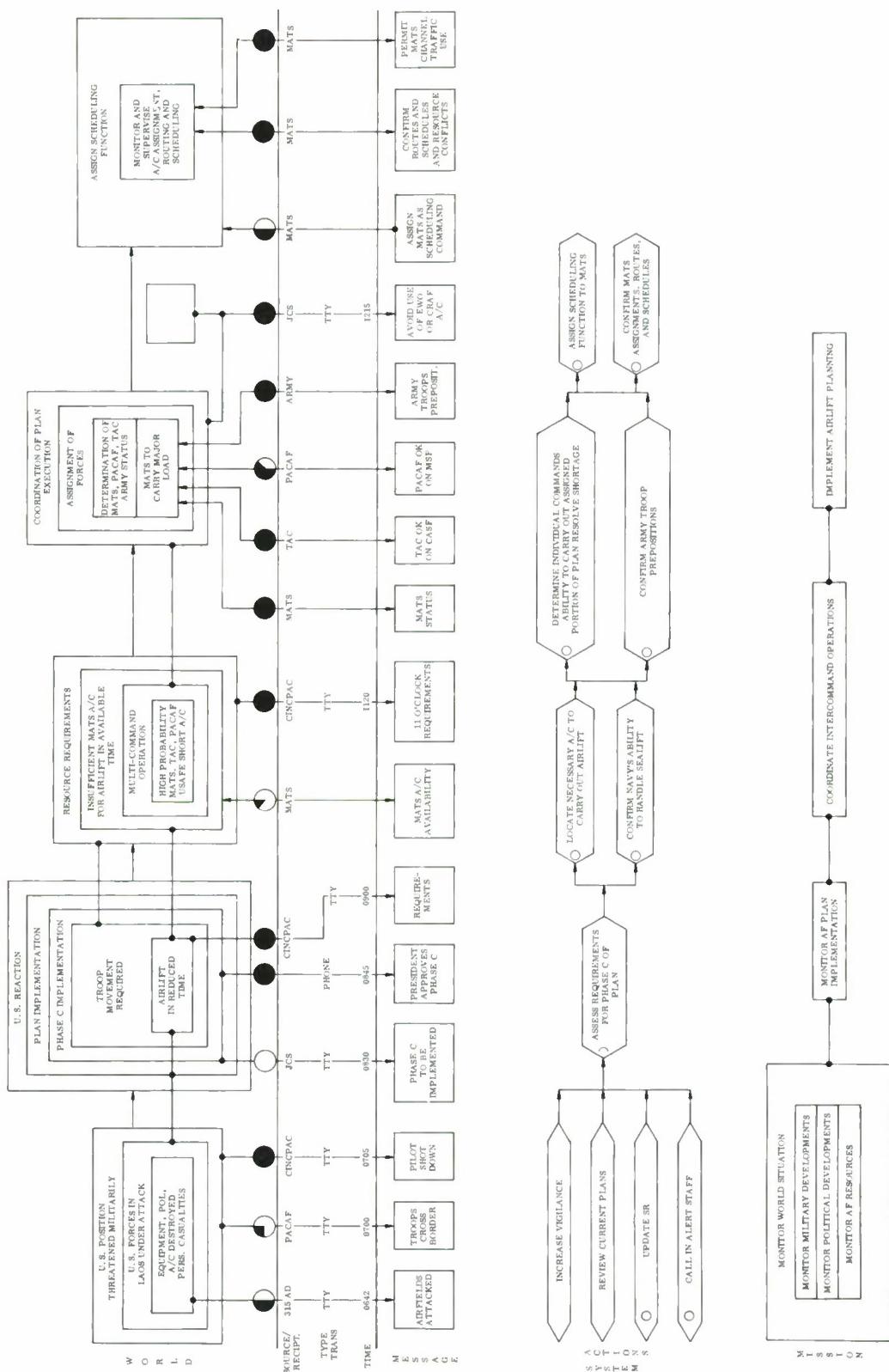
Figure A-5. Sample of a Flow Diagram

① ②

| BLOCKADE OF BERLIN |
| Convoys Stopped |
| Convoy A Stopped | Convoy B Stopped |

Message 1   Message 2   Message 3
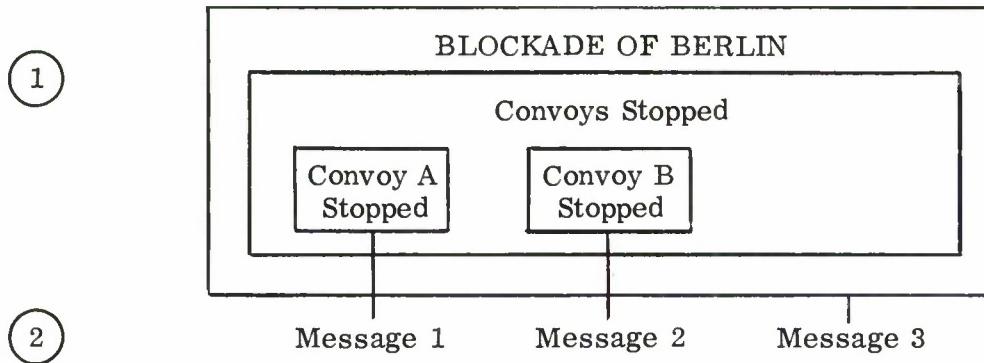
Figure A-6.   Sample of "Specification" in an Exercise Flow Diagram

information to a system with a mission like that of 473L.   The more specific fact
that "POL has been destroyed" suggests that it may be necessary to replace this
loss, and this does suggest that there may be an Air Force mission involved.

A directed line between two facts indicates that, given the first fact, the second
is possible (or even probable).   It may be desirable to indicate how probable; this
may be done by either using numbers between 0 and 1 to indicate rough probability
values, or by writing phrases such as "likely" or "very likely" (as illustrated in
Figure A-3).

Bi-directional lines (◄───►) may be used to indicate that one state of affairs
will occur if, and only if, another occurs.

As shown in Figure A-7, the notations explained above are used in the bottom
part of the EFD chart to indicate the relationships between activators.

| Monitor Touchy Situations | | Prepare Airlift Schedules When They are Needed |
| Monitor Berlin Situations | ───► | |
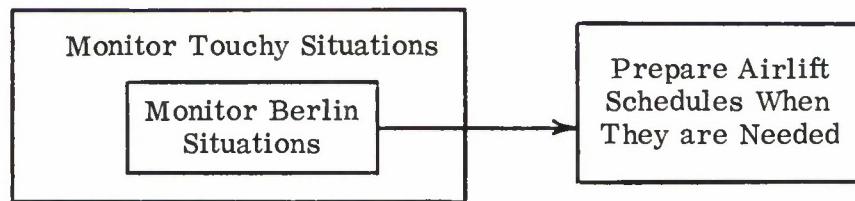
Figure A-7.   Logic of Activators

Message contents, the mode of transmission (teletype or telephone), and time of transmission can be stated briefly along the message row, which lies between Parts 1 and 2. An arrowhead at the end of a line representing a message indicates the direction of flow. Messages may go either into the system, out of the system, or both as in the case of telephone conversations. In a telephone conversation, the originator (system or exercise controller) is indicated by filling in the arrowhead on the side of the originator.

System actions are represented in the middle part of the chart by means of boxes with the following shape: ⟨⎯⎯⎯⎯⟩ . The same kind of connectors can be used with these kinds of boxes. In Figure A-6, the only connector used in the middle of the chart is the connector representing implication, and this connector is suppressed in order to simplify reading the chart. The connection between a fact communicated by a message and the action which it is intended to initiate is left implicit, being represented merely by locating the fact over the action which it is intended to create.

Some of the FCE technique notation shown in Table 1, p. 6, may prove useful where the structure is more complex than that in Figure A-6. For this reason, some FCE notation is included in the EFD notation which is summarized in Table A-1, p. A-2. For example, this FCE notation is helpful for indicating when the receipt of two messages by the system is needed for it to infer a fact, or when either one of two messages is sufficient. (See Figure A-8.)
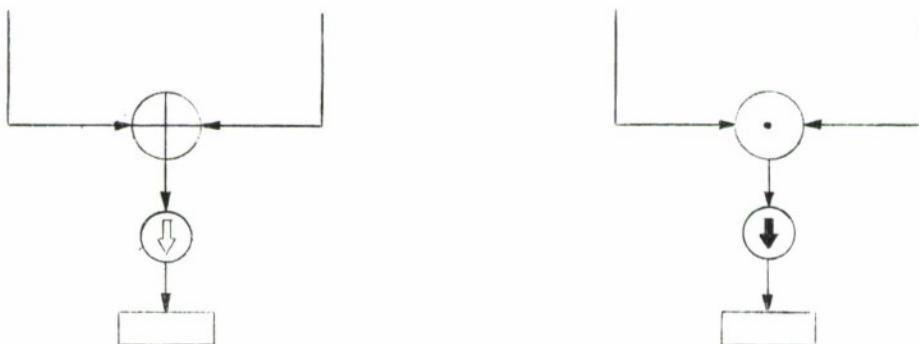
Receipt of Two Messages Required to Infer a Fact

Receipt of Either One of Two Messages is Sufficient for System to Infer a Fact

Figure A-8. Example of Uses of the FCE Notation

System actions that are clearly observable are indicated in an EFD by placing a circle inside the box representing that action. Actions that may or may not be observable are represented by placing half circles in the box which represents them. An example of the latter in Figure A-6 is the box containing the requirements for phase R of the plan. The act of obtaining this information may or may not be observable; a person may perform it completely in his head or he may say to others, "What are the requirements for phase R of the plan?" However, failure to observe the latter action does not indicate that the system has not performed the action indicated.

APPLICATIONS

DESIGN AND CONSTRUCTION

In the design of an exercise, an EFD may make it simpler to choose certain exercise parameters rationally because it makes explicit the structure of an exercise being designed. For example, the structure indicated by the part of the diagram describing the scenario shows how complicated the logical structure of the scenario is. An EFD allows this logical complexity of an exercise plan to be compared to that of other exercises so that an exercise of either more or less complexity can be selected as desired.

A major use of the EFD technique may be to provide a means of communication between personnel involved in the design, construction, and execution of exercises. Designers can use a diagram to communicate their intentions to those involved in the construction of an exercise, and constructors can use it to communicate their intentions to those who run the exercise.

The EFD technique can be used to check whether the exercise as written actually does what was intended by the designers. Thus, the person looking over the messages and drawing a diagram after the messages have been produced can try to diagram what he thinks the exercise actually looks like. This can be compared to the initial design diagram, and any differences between the initial and the final diagram would constitute errors in the construction. The exercise flow diagram technique may also serve as a checking technique because it may show where things have been left out.

MONITORING

A formalized EFD provides a technique for use in monitoring exercises by providing a standardized notation for indicating the behavior to be looked for. By making explicit the reasons why the systems should act in the way that the exercise designers have suggested, an EFD also provides a technique for use in controlling the behavior of the system.

DEBRIEFING AND EVALUATION

The same features which made the exercise flow diagram useful in exercise design and construction may also be helpful in debriefing and evaluation. The exercise flow diagram provides a graphic description of an exercise shorn of particular details. Thus it (a) simplifies the task of explaining exactly what the system ought to have done and (b) indicates the inputs that the exercisers gave to the system in order to get it done.

One can consider an exercise evaluation as a determination of system capabilities in certain directions. An exercise consists of presenting certain inputs to a system over time and observing the system's response to these inputs. The place along a series of inputs at which a response occurs can indicate what degree of capability the system has along a certain dimension. The EFD technique seems particularly well suited to determine system capabilities along the following dimensions:[*]

1. <u>Sensitivity</u> — The sensitivity of a system is its ability to determine the existence of a state of affairs from messages (or sensor inputs) which are less than completely obvious. For example, a system which becomes aware of an approaching aircraft when it is 1000 miles away may be said to be more sensitive than one which becomes aware of that aircraft 500 miles away; one that becomes aware of a state of affairs indicated to it in a routine message may be said to be more sensitive than one which becomes aware of it only when it is indicated by an emergency message. The facts stated in the "message row" ((1a)) are important inputs for an evaluation of system sensitivity.

---

[*] Readers who are accustomed to other meanings for the five terms defined here may wish to substitute other words for the names of these dimensions.

2.   Projectivity — The projectivity of a system is its ability to determine states of affairs of which it has not been directly informed. (Where these states of affairs are future events, the system's ability to do this is its ability to predict). Once a system becomes aware of some states of affairs, it may realize that these imply others, and this capability may be important in the system. Thus the system which projects a single convoy stoppage into a guess that a blockade will occur might be said to be more projective than one which has to wait until a blockade is announced. (Note that neither high sensitivity nor high projectivity is necessarily good.)

3.   Initiative — The initiative of a system is its awareness of its own mission and its willingness to perform it. (The two are hard to separate in practice.) This is often measured by determining which activator is required before a response occurs. A system which, on its own, realizes that it should do something is said to have more initiative than one which does nothing until ordered.

4.   Planning Ability — The planning ability of a system is its ability to prepare a scheme of action and to communicate this scheme to those system components that must execute it.

5.   Execution Ability — The execution ability of a system is its ability to carry out plans.

In the kind of exercise with which we are concerned, one does not determine system capabilities in these five directions individually. An exercise is a series of problems, each of which calls upon different capabilities in differing amounts. In a well-designed exercise, the problem solution is an observable piece of behavior. One purpose of exercise monitoring is to record when this behavior occurs. By applying elementary techniques of factor analysis to data describing both how much capability was required in each direction and how well the system did with each problem, one might be able to determine, roughly, the amount of capability demonstrated in each direction. (See Figure A-9.)

The results of such an analysis indicate the degree of sensitivity, initiative, and the like, of the system. The degrees are not numerical, but they can be compared intuitively with the requirements placed on the system by its environment. For example, an exercise might show that a system for detecting a missile attack is
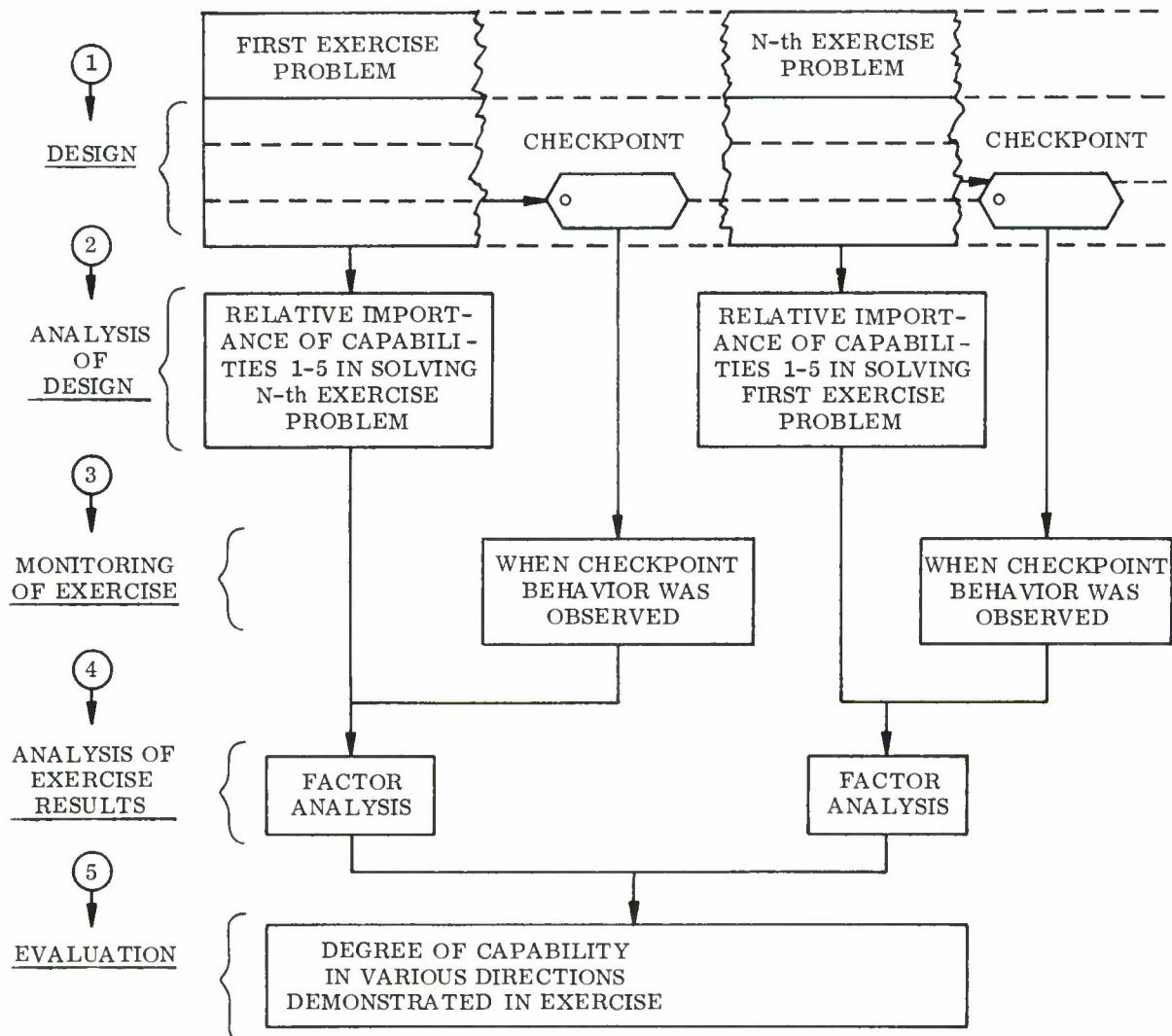
Figure A-9. Steps Toward Using an EFD in Multi-Directional System Analysis

sufficiently sensitive to detect a missile attack by ten or more missiles. If an attack by less than ten missiles is likely, then the system would be said to be insufficiently sensitive. On the other hand, if the indicated sensitivity could lead to false alarms, then the system would be said to be too sensitive. Thus, the characteristics demonstrated in an exercise might be compared with a description of the anticipated environment to determine if the system is adequate to that environment, and to determine in which directions improvements might be sought.

PREDICTIONS AND AUTOMATION

The structure represented by an exercise flow diagram (without any captions in the boxes) represents one aspect of exercise structure. The EFD of an exercise with different captions in the boxes may provide designs for replications of exercises. Such designs for replications predict that system behavior will be roughly the same in each replication (if we disregard transfer of learning).

A second, more sophisticated kind of prediction, for which an EFD might provide a basis, is prediction based on certain measurable characteristics of the EFD representing an exercise. Thus the "complexity" of an exercise might be defined as a function of the depth of nesting in the "world," or the number of inferences possible. The "difficulty" of an exercise might be a function of the amount of shading in message circles. These measures might be used to predict that a system would perform similarly (in some respect) in exercises having similar degrees of difficulty, complexity, or other properties.

The EFD technique may also be used to help automate the design and control of exercises. One might write computer programs to provide exercise flow diagrams. The input of such a program would specify the desired complexity of the exercise; the output would be the flow diagram for an exercise meeting these specifications. One could program a computer to put captions in the boxes of the flow diagrams it writes, but such programming might be quite time consuming.

The EFD technique might also provide a basis for automating exercise control. The EFD representation of an exercise, rewritten in the linear notation of symbolic logic, provides a basic scheme for organizing control messages for automated retrieval. When anticipated behavior does not occur, an algorithm can work backwards

from the representation of that behavior in a flow diagram and find which of the steps leading up to the expected behavior has not occurred. Prepared messages strengthening the system inputs which were to have led to the unperformed behavior can then be presented to the system to produce the behavior. Similar techniques can be used when the system does something that was not expected.

RECOMMENDATIONS

The EFD technique remains untried in actual use. Several things remain to be done:

1. Specification—Before the EFD technique can be applied to a particular system, certain details of the technique must be spelled out. For example, one may assign different meanings to the shading of the message circles (see Table A-1, p. A-2, for different systems).

2. Test of utility—The best way to evaluate the technique is to use it and to attempt to determine whether it (1) improves the quality of exercises, (2) cuts the cost (in time and effort) of constructing an exercise, (3) provides the basis for a good debriefing tool, and (4) provides the basis for better evaluation of exercise results.

3. Test of reliability—A technique is said to be reliable if it continues to give similar results in similar situations. In the EFD technique, this means two different things: (1) Given the same EFD representation of an exercise, do several people produce roughly comparable exercises when asked to expand on it? (2) Given the same exercise to diagram, do different people produce roughly comparable flow diagrams?

4. Validation—Efforts might be made to validate the technique experimentally to determine whether: (1) It is a medium for the replication of exercises, producing comparable behavior in response to several exercises based on the same EFD structure; (2) It provides a medium for predicting system behavior and thus for determining system capabilities on the basis of exercise results; (3) The kinds of predictions made are actually useful to a system user or to those who make design recommendations.

5. <u>Extension</u> — One might study the EFD representation of exercises rather than the exercise being represented. Such a study might result in definitions of intuitive notions, such as "exercise complexity," in terms of objective characteristics of the EFD, such as depth of nesting.

6. <u>Applications to automation</u> — One might try using the technique as the basis for the automation of some exercise functions including design, construction, and monitoring in the manner suggested briefly on p. A-17.

APPENDIX B

THE RESOURCE ASSIGNMENT (RA) MODEL

APPENDIX B

THE RESOURCE ASSIGNMENT (RA) MODEL

INTRODUCTION

The resource assignment (RA) model describes the information processing that leads from the statement of a resource allocation problem to its solution. The machinery used is largely that of mathematical logic. The model is similar to the general problem solver of Newell, Shaw, and Simon[1] in many respects. The RA model can be used to describe various elements involved in problem solving and appears to have a number of applications in system definition, problem selection, exercise definition, system evaluation, and system design.

SUMMARY OF APPLICATIONS

A command and control system supports a commander (or group of commanders) whose task it is to detect and to solve problems in his area of command. The RA model describes the general activity of problem solving and thus, in some sense, what command and control systems do. A command and control system is finally evaluated in terms of its ability to do its job, rather than in terms of the qualities of the individual elements comprising the system.

By describing a problem precisely and the elements that go into its statement and solution, the RA model provides a medium for functionally defining the system being evaluated. This functional description defines the class of things with which the system should be able to deal and thus the class from which an exercise problem can be drawn. By describing the elements that influence system behavior it can contribute to the design and control of an exercise, both of which aim to shape this behavior. Because it describes the interrelationships between the problem and its solution(s), it may prove useful in the evaluation of exercise results and in making design recommendations based on them.

BRIEF DESCRIPTION OF MODEL

The elements of a RA model are a series of known facts described precisely in a formal language; a series of rules for manipulating these items; and a description

of the conditions defining solutions to the problem. A problem is solved by manipulating the given facts according to the rules to yield something that meets the conditions of a solution.

Such a description of a resource assignment problem is much like the standard descriptions of board games. The given conditions are the conditions of the pieces on the board when the game starts. The manipulations allowed are those defined by the rules of the game, and the conditions which constitute a solution correspond to winning the game (getting a checkmate in chess or clearing the board of the opponent's pieces in checkers). However, resource assignment problems are generally more complex than games, and the rules are generally looser.

BASIC NOTIONS

THE NATURE OF COMMAND AND CONTROL SYSTEMS

Both the inputs and the outputs of a command and control system are information. Examples of other information processing systems are digital computers, human brains, newspapers, and some industrial control subsystems. These differ from systems whose inputs and outputs are energy (such as engines, muscles, and generators) and those with material inputs and outputs (such as factories, mines, and digestive systems). In the real world, none of these types of systems exists since the inputs and outputs of every real system combine information, energy, and materials. However, by looking at command and control systems as information transformation systems, it is possible to say almost everything in which we are interested.

Information systems have power because energy and material transformers act on the information produced by the information systems. In the extreme case where the form of the information itself is binding (such as a thermostat or governor), the energy transformer has little choice. In the systems with which we are concerned, the interrelations between the energy-matter transformers and the information transformer are considerably more complex.

Information systems differ among themselves with respect to the kind of information they take in and put out and the environment in which they operate. The

command and control system exists in military command structure and issues orders (or provides information to those who issue orders) within the command structure. The nature of these orders and the mode of response to them are important in defining the system because its "powers" are restricted by this command structure. However, this is a two-way relationship. The existence of the system at a certain point in the command structure will also tend to influence that structure. This two-way relationship is important in system design because it means that the existence of the system changes the environment.

The range of inputs and outputs of a particular command and control system, together with its environment, will tend to define it. In such systems, the inputs are information about the status of some aspect of the universe. Some of this information is about the system being controlled; some is about the environment of that system; some deals with the interrelationship between these two; and some is about the status of the command and control system itself. The outputs of such systems are usually information that can affect only the system being controlled.

Usually it will be fairly difficult to draw a precise line between the command and control system and the system which it is controlling, because it is difficult to draw a precise line between what is information and what is energy or matter. Considering the human mind, one might say that the mind sends information to the mouth, which transforms that information into speech, but one might also say that speech itself is information that transforms the action of others. Every piece of information is partly physical, and everything physical contains some information. Therefore, one can view all the examples of information systems presented above as energy or matter transformers.

A military commander knows something about the state into which he would like to put (or to maintain) the universe of which he is a part. The universe can be put into this state only by means of the information the commander outputs. See Figure B-1.

This information can affect only those parts of the world within the authority of the commander (his command). Since the process of getting the information to the controlled system and getting the controlled system to act on the external world takes time, the commander must consider the nature of the command structure and
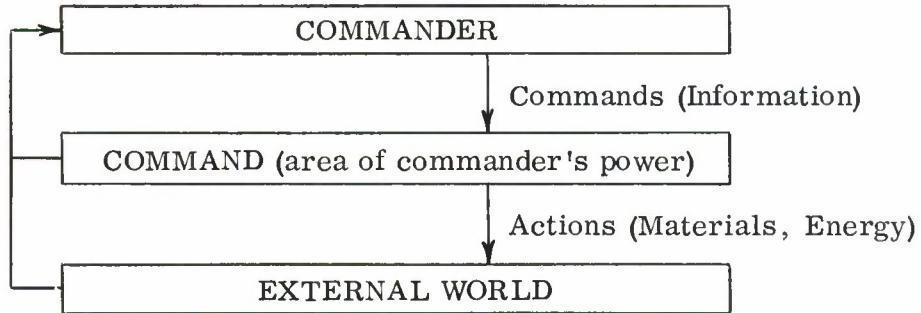
Figure B-1. The Commander and His Environment (Simplified Diagram)

the nature of the external world in his attempt to predict the effects that will come about as a result of the information he outputs. During this time (or possibly afterwards), the external world can affect his command by means of his actions or reactions. Finally, the information coming to the commander is affected by the characteristics of information transmission and other processing systems between the source and receiver. In the real world, the shortcomings of these may degrade the commander's capabilities. These considerations yield the more complex structure shown in Figure B-2.
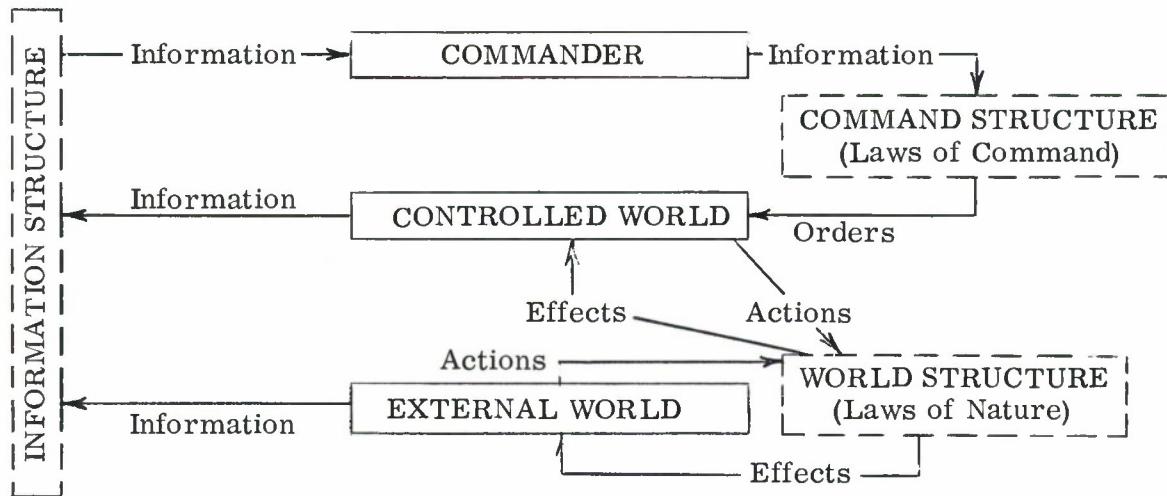
Figure B-2. The Commander and His Environment

One way of dealing with such systems is as follows. One first describes the behavior of an idealized command and control system having complete information. It can examine all this information instantaneously, consider every possible course of action, and select the best course of action from the alternatives. It will have clear-cut criteria for such choices, and it will not make mistakes. In terms of such an idealized device we can then define categories such as "a problem," "a resource," and "an allocation of the resource for solution of a problem."

When a system does not have all conceivable information, adding information via messages will make the system approach the state of the idealized device. Comparing the system before and after receiving such messages with the ideal device (towards which the system aims) will provide a non-numerical measure of the information content of the message in the particular context. Such a measure will be a function of the state of the receiver, which differs from Shannon's[2] measure. Shannon's measure is a function of the unexpectedness of the message relative to the receiver. The measure outlined here will be unprecise, but it will be a measure of the utility of the message relative to the aim of the receiver. The difference may be indicated by the observation that the message can be very surprising to a receiver without necessarily being useful to him.

Since real systems cannot search all possible alternate solutions, such systems must search the areas which seem to be most promising or those in which solutions are believed to be most thickly distributed. The inability to order alternatives in a completely specified, everywhere comparable hierarchy will force the system to make decisions.

Because the real system is not homogeneous (it consists of parts with different capabilities), it is forced to use allocation strategies to divide work among its various parts. Where the parts are men and machines such strategies are involved in "man-machine trade-offs." Where the parts are human beings the limitations of certain parts are what is involved in "human factors." The efficiency with which these allocation problems are handled in alternative designs can be elucidated by comparing them to the ideal.

The fiction of this idealized device will permit us to classify all conceivable operations an ideal command and control system could perform. It provides an

information "space" for which the information system involved in a particular command and control system provides a metric. It is advantageous to consider the idealized device first because it provides a standard against which to measure the various shortcomings that arise and the value of efforts to overcome them. Such a device is one which can be degraded in one direction without adding the complications of degradation in other directions. This enables one to deal with complex combinations of problems in the smaller sized parts of which they are composed.

The ideal state does not contain any of the real problems of command and control systems. It acts as a neutral, sterile culture-medium in which the individual bacteria that affect the command and control system's body can be isolated and studied separately. The theory to be developed will be as precise as feasible, but it will not be quantitative. It will be developed in a form that will make it possible to formalize it, but not necessarily to assign a satisfactory metric. This means that the values assigned to descriptions of elements such as "difficulty" of decisions, "complexity" of problems, and "amounts" of information may not correspond to the real numbers. Therefore, the mathematics used in such a theory will have to be that which can deal with non-numerical objects.

WHAT A COMMAND AND CONTROL SYSTEM DOES

The commander can change the world in which he exists by ordering certain transformations of that world. The job of a command and control system is to help him to decide what transformations to order. Insofar as this decision is an information processing activity, it is similar to the solution of any problem.

Problems

A problem is the input to an information processor, and thus it is a piece of information. A problem describes a problem state. We will not distinguish between the problem state and its description.

A problem consists of four parts:

1.   An initial state—the state of affairs existing when the problem begins. In a chess problem this is a distribution of pieces on a board. In a resource allocation problem this is the distribution of the resources available for the solution of the problem.

2.   The final state—the state of affairs it is desired to reach. Determining how to reach this state constitutes the solution of the problem. The final state need not be unique, nor need there be only one way to reach a given final state. (A problem may have a number of solutions.) And the final state need not be described in full detail. One may be given only certain criteria for the final state. In a chess problem the final state is usually a checkmate. Many positions constitute checkmates, and there are many ways of getting into each of those positions. In a resource allocation problem, the final state is an optimal allocation of the available resources. Here an important difference between a resource allocation problem and a chess problem becomes apparent. The solution of a resource allocation problem may require judgment to determine what constitutes an optimal allocation of the resources, whereas in a chess problem all solutions are equivalent.

3.   A set of allowable transformations of state—the operations one can perform on the description of the initial state to change it, through a series of intermediate states, into a description of the final state. In a chess problem this is the set of allowable moves of chess. In a resource allocation problem this is the assignment of resources to missions, moving the resources, and other moves.

4.   A set of restrictions—descriptions of intermediate states that are not allowed to exist, even though they may be reached by the transformations of part 3. In a chess problem such restrictions prevent one from getting checkmated oneself in the process of checkmating one's opponent. In a resource allocation problem the restrictions indicate items such as territories over which flight is forbidden.

These elements are illustrated in the simplified maze-problem shown in Figure B-3. (It is assumed that the maze is so narrow that the solver cannot turn around.)
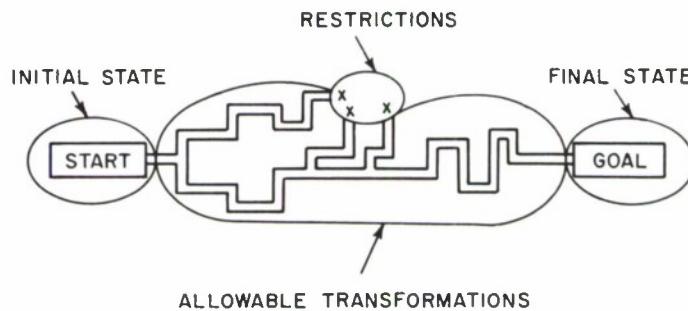
Figure B-3. The Four Parts of a Sample Problem

These four elements of a problem are relatively objective. They do not depend on the system's solving the problem, although the ability of the system to represent these elements may be important.

Two more elements to be mentioned later in this discussion are described below. We will not include these in our model, but they are important in using the model for controlling an exercise and for evaluating its results.

1. The solver's awareness of the elements of the problem corresponds roughly to how he represents (to himself) the elements of the problem description. This awareness is particularly important when a large part of the problem difficulty consists of defining the problem or where the potential solver's orientation (set) is fundamentally wrong. Many familiar puzzles depend on precisely this element.

2. The solver's notion of his role in the problem is important when the solver does not realize that it is his job to solve the problem. This notion of his role is very important when other objective aspects of the problem do not move a given problem solver toward the solution of the problem. Therefore, this item is important in using this model for exercise control.

Before discussing a problem and its solution we need to define some further terminology (illustrated in Figures B-4 through B-7):

Step: single application of a single transformation rule either to the initial state or to some intermediate state.

Potential solution: any sequence of zero or more steps not leading to a state that violates a restriction.

Solution: potential solution that leads to a final state.

Solution space: set of all solutions.

Problem space: set of all possible solutions. This includes the initial state and the solution space. One measure of the difficulty of the problem is the proportion of problem space occupied by solution space.

Subset-heuristic: partitioning the problem space into two or more disjoint parts, ordered according to decreasing density of solutions in the parts. Successful searches of the problem space are more likely when one searches the subspaces in this order.
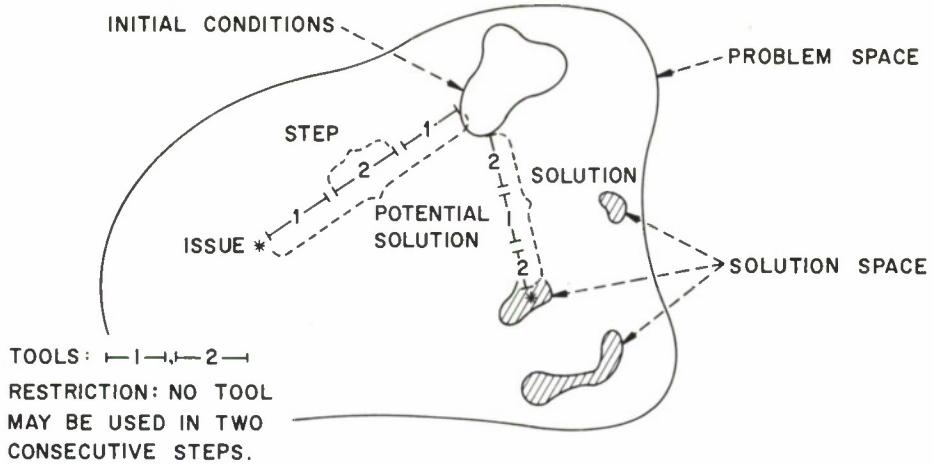
Figure B-4. Sample Anatomy of a Problem
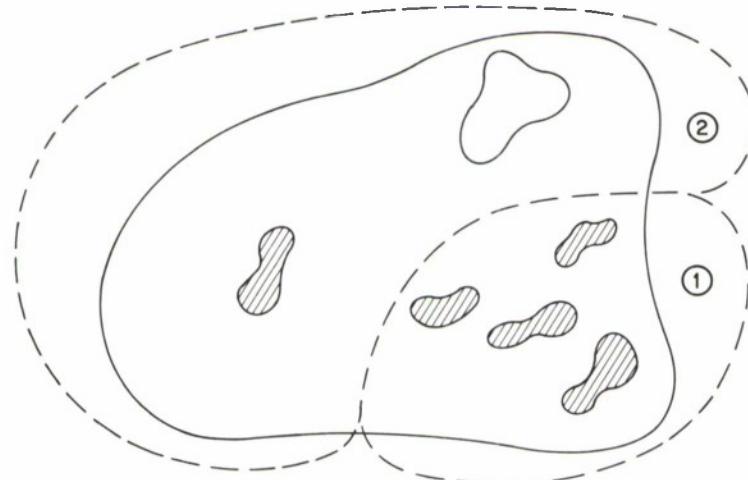
Figure B-5. Dotted Lines Are a Subset-Heuristic
(Using Notation Similar to Previous Figure)
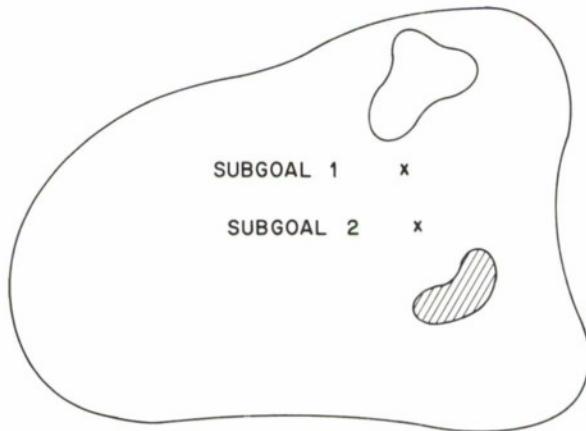
Figure B-6. Subgoals

Algorithm: search through the problem space with each step clearly defined and with the solution guaranteed after a finite number of steps.

British Museum algorithm: algorithm whose success is guaranteed because of the finitude of the problem space. Its ordering depends on the notational features of the language describing the problem space.

Issue: state that results from applying any potential solution to the initial state.

Subgoal: pseudo-final state known to be a suitable intermediate state in the achievement of the final state. Ways of selecting such subgoals are discussed by Newell, Shaw, and Simon.[1]

Subgoal heuristic: separating a problem into subproblems, where the final state of each subproblem is a subgoal. This can improve the efficiency of one's search. The difference between this kind of heuristic and a subset heuristic is indicated in Figure B-7.
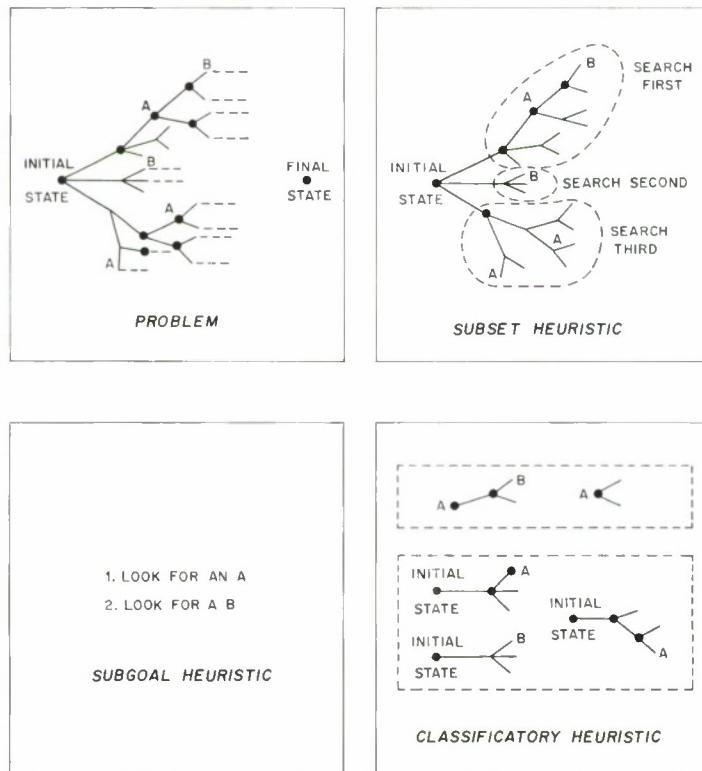
Figure B-7. Three Kinds of Heuristics

<u>Classificatory heuristic</u>: organizing one's tools more efficiently. Such organization will be useful when (a) one has a large period of time during which one can prepare to solve a problem and (b) one has much of the information one is to use on hand before one is called upon to solve the problem. (Such situations appear frequently in command and control systems, and therefore this kind of heuristic may prove useful in making design recommendations for the organization of the system's data base.) Classificatory heuristics can be divided into two types. "Rearranging heuristics" involve reordering one's basic set of tools and restrictions so that searches become more profitable. "Selecting heuristics" are procedures based on selecting a subset of the available tools in order to limit the area of one's search through possible combinations of such tools (see Figure B-8).

TOOLS AS GIVEN

$T_1, T_2, \ldots, T_n$

REARRANGING

$T_1, T_3, \ldots, T_2, T_4 \cdots$

SELECTING

$T_1, T_3, \ldots T_n$

Figure B-8. Two Types of Classificatory Heuristics

A kind of heuristic to which we will not assign a name, and which is closely related to selecting heuristics, is a technique that depends on placing more restrictions on the allowable solutions. Although such restrictions make the problem appear more difficult, they may actually help solve the problem by limiting the number of solutions that must be examined. In the course of an exercise, the problem being solved by a system becomes increasingly specified by information provided by messages, or other inputs. This increasing specification of details changes the requirements placed on the system seeking to solve the problem.

If we proceed from the most narrowly defined and fully specified kind of problem to the least narrow and least fully specified type, we first come upon what might be called a "task." A <u>task</u> is defined as a problem in which the steps for solution are completely specified. A "solution" consists of carrying out these steps. The second kind of problem is a "puzzle." A <u>puzzle</u> is a problem in which it is known that a solution can be obtained, given the tools and restrictions, and for which there is only one solution or all solutions are of equal value. In this appendix we shall be concerned primarily with puzzles. The problems presented to command and control systems are somewhat more complex, but the puzzle constitutes the core of the exercise.

There will be puzzles where the initial state is not completely specified and where at least part of the solver's problem is to specify that initial state. Similarly, there will be puzzles where the set of tools allowed for solutions is not completely specified. And there will be puzzles where there are either several solutions of different value or no completely satisfactory solutions so that the least unsatisfactory must be chosen.

Puzzles[*]

A puzzle is a basic problem that includes only those factors needed for the solution. The puzzle does not contain elements injected for consistency, for realism, and so forth. It does not contain background information; it does not contain the information in messages not germane to the problem to be solved; it does not contain any information the system may receive that suggests it may be dealing with a different problem; it does not contain any information not established with certainty.

The puzzles that comprise an exercise form a series such that the solution of one becomes the initial condition of the next. (This structure is depicted in Figure B-9.) The series of puzzles defines the normative path[6] of the exercise. Puzzles are distinguished from the more complex problems within which they are embedded.

PUZZLE 1	SOLUTION 1	PUZZLE 2	SOLUTION P
Initial conditions$_1$ Tools$_1$ Restrictions$_1$ Final Conditions$_1$	The steps which comprise solution 1 use all of the initial conditions$_1$, tools$_1$, restrictions$_1$, and yield the unique issue satisfying final conditions$_1$.	Initial Conditions$_2$ (= the issue of Solution 1) Tools$_2$ Restrictions$_2$ Final Conditions$_2$	The issue of this solution meets final conditions of the problem.

Figure B-9. Puzzles in an Exercise

---

[*]This section is taken from our Concept Paper No. 5.

In the first puzzle of an exercise, the initial condition is usually a group of people in the system. The final condition is usually the recognition of the problem by this group and the definition of it. That is, the targets of the first problem are:

1. Awareness of the target condition. (Note that awareness of the target condition and the target condition itself are two totally different things.)

2. Awareness that the job of the system is getting the Air Force into the target condition.

3. A definition of the allowable tools.

4. A definition of the restrictions.

There is seldom any observable behavior that can tell us whether the system has achieved this target condition. Some person may remark that he thinks there is a problem and what he thinks that problem is, but if nobody says anything, there is no way of telling whether the system has solved its first problem.

In general, it will be difficult to specify the set of tools or restrictions. In the statement of a problem to a system, the tools and restrictions will seldom be stated explicitly. Essentially, the tools are assumed to be "common sense," and the restrictions are assumed to be "good sense." These tools and restrictions can be specified, but it is not generally necessary to specify them if the solution of the problem is to be carried out by a person. It is important to specify them if the solution is to be carried out by a machine. (If there is difficulty in specifying the tools to be used in a solution step, this fact often suggests that the function performed is to be a human rather than machine function.)

Three situations may increase the difficulty of a puzzle:

1. The puzzle may be embedded in a situation of similar puzzles the system must consider but are not germane to later puzzles.

2. The system may not see its mission clearly, or the system may not accept the premise that its mission is as outlined in the puzzle description. Even where the mission is stated explicitly at the start of the exercise, the mission

statement may not come to the mind of the players at the right time. The mission of the system and the requirements must be brought together in some mind at the same time, or the mission can be made explicit in system procedures that might require checking of any incoming requirements. One output of an exercise might be the observation that system procedures are not adequate since they do not guarantee problem awareness by requiring steps that will automatically get the system into motion toward fulfilling its mission. (Perhaps no one was assigned to check the consistency of requirements and plans.) This shortcoming would define a necessary system function as a result of an analysis of the system mission. A more general statement of the cause of this increased difficulty would be to say that it was difficult for the system to establish part of the initial condition. The establishment of such an initial condition is a problem itself. It differs from a simple puzzle in the sense that the target condition is defined only after solving the second puzzle (whose initial condition is the final condition of the first puzzle). It almost appears that this kind of a problem is such that the system cannot define it before solving it. Thus, the system must work on several problems at once since it does not know which is its real problem.

3. The system may not be aware of the problem of monitoring the situation to find puzzles. That is, the system may not be motivated to become aware of puzzles. There may be no one whose job it is to become aware of them or to arouse the rest of the system. If the system does not become aware of a puzzle it cannot solve it.

ELEMENTS OF THE MODEL

We have attempted to justify describing the job of a command and control system as a job of symbol manipulation and to justify basic categories for describing this job. Symbol manipulation can be described as precisely as any other phenomenon. Since the work of Frege,[3] increasingly sophisticated techniques for such description have become available. The bases for all such techniques are two guiding principles: (a) all manipulations of symbols are to be based on the form of the symbols and not their meanings, and (b) all allowable manipulations are described by precisely stated rules whose results, when applied to any given string

of symbols, are unequivocal. In general, the elements of such a theory are:
(a) an alphabet of symbols, usually finite, (b) a set of sentences or well-formed
formulas (a primitive recursive subset of the concatenation semigroup generated
by (a)), and (c) rules (usually general recursive) for manipulating the elements of
(b) and often for dividing them into subsets.

The rest of this appendix describes the elements for constructing resource
allocation models for specific applications. The basic building blocks are pre-
sented informally rather than by a formal, complete model for two reasons. A
full-fledged formalism with all the details required for a general purpose model
would tend to be difficult to follow and uninteresting. Also, those who apply this
model to particular problem areas will want to change some details in order to
simplify the generalized model for their specific applications.

NOTATION

The resource allocation model represents the information processing that
might go on in the course of problem solving in a way that is fairly independent of
system configuration. The basic elements in the model are objects and their
properties. Objects may be concrete things such as resources; or they may be
abstract items such as plans, missions, utilities, and assignments. Objects will
have properties such as range, importance, and commands. (The command to
which an assignment is made is treated as a property of the assignment.)

Properties, in turn, will have values (such as 1000 miles, 3, or PACAF).
The names of properties will be written in parentheses following the name of the
object; the value will follow the name of the property (within the parentheses),
separated from it by a comma. Capital letters will denote objects and variables
ranging over objects; lower case letters will denote properties, their values, and
variables ranging over properties and values.

In the full formalism, the name of an object would be treated as a property of
the object. Objects would be totally property-less "pincushions" into which the

properties were inserted. In this notation,[*]

$$R_4(\text{name, otis airforce base})(\text{ceiling, 1000 ft}) \qquad (1)$$

indicates that the ceiling at Otis Air Force Base is 1000 ft.

In the kinds of applications of interest here, this type of notation avoids much redundancy. We shall distinguish[†] between items describing details of a particular problem and statements which will be true independently of the specific problem. The latter statements may be viewed as the space within which one can maneuver while trying to solve the specific problem. The characteristics of this space will usually be stored in tables or mathematical functions, which are essentially infinitely large tables.

The name of a part may provide information about it. For example, if one is given only the tail number of an aircraft, by referring to tables one can determine many properties of the aircraft. Tables might show an aircraft whose tail number was 1234 to be a member of a class called "CR-C-130." The property of having a certain rate of climb at a certain altitude and with a certain load may be derived from the name of the object. Since this derivation can be generalized easily, these properties need not be repeated in describing each individual resource.

The notation of (1) can be written more clearly as

$$\text{OTIS AFB (ceiling, 1000 ft)} \qquad (1')$$

Instead of variables, in this appendix we will often use resource names.

SAMPLE OF A SIMPLIFIED PROBLEM

For purposes of illustration, consider the simplified airlift problem shown in Figure B-10. Assume that we are required to move a aircraft from point X to

---

[*]Reminiscent of Quine's elimination of individuals (Ref. 4).

[†]The reasons for making this distinction are detailed on p. B-20.

INITIAL STATE

INTERMEDIATE STATE

FINAL STATE
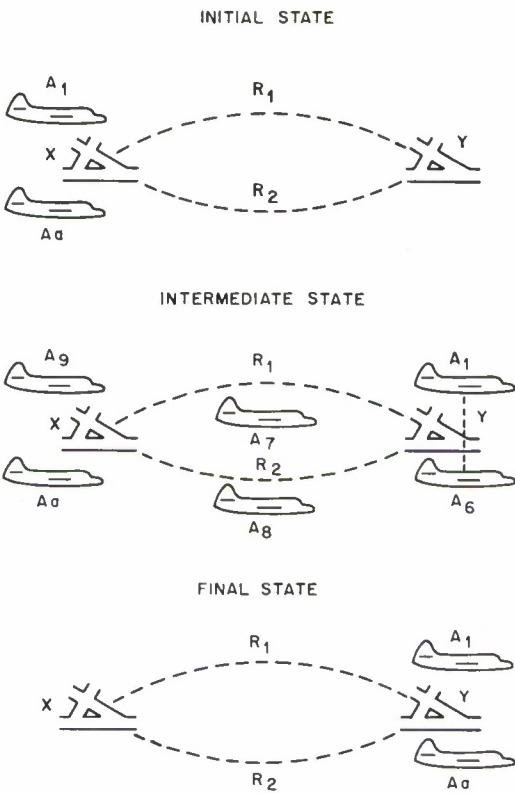
Figure B-10. A Simplified Airlift Problem

point Y over routes R_1 and R_2. Assume, also, that the aircraft must be spaced 1 min apart, that it takes i hours to traverse route R_i. Using the notation of (1'), one can express a part of the initial conditions as

$$A_1(\text{location, x})(\text{time, t}),\ldots, A_a(\text{location, x})(\text{time, t}) \tag{2}$$

where t is a parameter in terms of which the solution will be stated.

Those familiar with symbolic logic will see that (2) can be shortened by using a bounded universal quantifier

$$\underset{i}{\overset{n}{(x)}}$$

meaning "for all x between 1 and n."

Thus (2) can be rewritten as

$$\overset{a}{\underset{1}{(i)}}\, A_i(\text{location, x})(\text{time, t})\,. \qquad (2')$$

Similarly, we can use Kleene's[5] least number operator

$$\mu y \quad (\text{or } \mu y_{t<z}) \qquad (3)$$

for "the least t such that" (or "the least t less than z such that") to denote the final state:

$$\mu t'\,\overset{a}{\underset{1}{(z)}}\, A_z(\text{location, x})(\text{time, t}') \qquad (3)$$

which says "the least t' such that all the aircraft $(A_1 \ldots A_a)$ are at location at time t'."

As (2') and (3) suggest, we can import many of the standard notational devices of mathematical logic in roles similar to their usual uses. Thus we shall use

p.s	for	p and s
pvs	for	p or s
~p	for	not p
cϵd	for	c is a member of d
(Ex)	for	there is an x .

We shall use this notation to produce complex properties as well as to produce sentential functions. For example, "red v white" will be used for the property of being either red or white, and "red . white" for the property of being red and white.

In the sample problem, assume that routes R_1 and R_2 are free—no other aircraft are using them:

$$\overset{2}{\underset{1}{(i)}}\,(t)\, R_i(\text{utilization at t, 0}) \qquad (4)$$

We have now defined the initial (1' and 4) and final (3) conditions of the problem.

To turn this static model into a dynamic one, we must write rules for trans- formations of states. An arrow will denote a change from one state to another:

$$\alpha \rightarrow \beta$$

Either α or β may be a blank space. The expression ($\rightarrow \beta$) means simply "write β"; ($\alpha \rightarrow$) means "erase α." A bidirectional arrow denotes "if and only if." Thus, ($\alpha \rightarrow \beta$) $\longleftrightarrow$ ($\alpha' \rightarrow \beta'$) means that a change of α to β can occur if and only if a change from α' to β' occurs. The statement is intended to include the assumption that both α and α' hold; α, α', β and β' may contain arrows themselves.

To write the transformation rules needed for our particular problem, we must introduce a device to indicate that a particular route R_i is free at time t and not free at some other time. The notation of (4) can be used to indicate that route R_i is empty at time t, where t is the beginning of a flight time:

$$R_i \text{ (utilization at t, 0)} . \tag{4'}$$

Using the notation already defined, we see that

$$A_i \text{ (assignment, } r_j \text{) (time, w)}$$

asserts that resource A_i has been assigned to route r_j at time w.

The assignment rules for our specific problem can be summarized by stating that R_i requires i hours to be traversed from X to Y, and that aircraft must be separated by 1 min. These statements can be written as: (h = hours, m = minutes)

$$A_i \text{ (location, x) (time, h:m)} \rightarrow A_i \text{ (location, y) (time, (h + j):m)} \longleftrightarrow$$
$$\rightarrow A_i \text{ (assignment, } r_j \text{) (time, h:m)} \tag{5}$$

$$A_i \text{ (location, x) (time, h:m)} \rightarrow A_i \text{ (assignment, } r_j \text{) (time, h:m)} \longleftrightarrow$$
$$R_j \text{ (utilization at h:m, 0)} \rightarrow R_j \text{ (utilization at h:m, 1)} \tag{6}$$

To generate all possible solutions, one examines the initial situation to determine whether any transformation rule can be applied to change it. Then one examines the set of all resulting situations and continues to apply transformation rules as long as possible. Given the situation as described thus far, we can use this procedure to produce an infinite number of solutions to a problem and an infinite number of impossible situations. For example, since we have not indicated that time does not go backwards, we will present as solutions, assignments that occurred before the problem was stated. This suggests that we need to specify additional limitations.

For instance, we will limit a day to 24 hr. This can be stated as

$$(\text{time}, x) \rightarrow (\text{time}, s) \leftrightarrow s = h{:}m \cdot h \leq 24 \cdot m \leq 60 \tag{7}$$

which allows t to be any time of day, expressed at 1 min intervals. Given rules (5) and (6), (7) implies that aircraft must be spaced at no less than 1 min intervals.

The restriction that time does not go backwards (and limiting ourselves to a single day) can be stated as

$$(\text{time}, x) \rightarrow (\text{time}, s) \leftrightarrow s = h{:}m \cdot h \leq 24 \cdot m \leq 60 \cdot s > t. \tag{8}$$

In applying this model, we shall wish to distinguish between restrictions that cannot be changed and restrictions that might be changed. (For example, in an emergency it might be possible to space the aircraft more closely. But s > t indicates that time cannot go backwards, and this restriction cannot be changed.[*]) To make this distinction we will treat restrictions and other transformation rules as objects subject to other rules.

To do this we assign names to rules, using the semicolon as a sign of definition. Thus, to assign the name Ru_1 to restriction (8), we write

$$\text{Ru}_1 ; t \rightarrow s \leftrightarrow s = h{:}m \cdot h \leq 24 \cdot m \leq 60 \cdot s > t. \tag{8'}$$

---

[*] This is not quite true. See p. B-21.

Two restrictions are imposed on the use of the semicolon and the abbreviations introduced through its use:

1. A rule cannot contain its own name.

2. No name can be used for two different rules in the same problem.

Rules about rules will often involve the variables in the rule being governed. Therefore, we allow the variables to be made explicit in the abbreviation, and we will interpret the transformation of an abbreviated name as a transformation of the full rule. Thus, the variable t can be made explicit in (8') by writing $Ru_1(t)$ for Ru_1:

$$Ru_1(t); t \rightarrow s \leftrightarrow s = h{:}m \cdot h \leq 24 \cdot m \leq 60 \cdot s > t \qquad (8")$$

We can use (8") to indicate that time can go backwards in situations where a higher authority orders an earlier starting time for a mission. This can be expressed as

$$\text{ORDER (originator, higher command) (recipient, xxxl system)}$$
$$\text{(stating, } Ru_1(t) \rightarrow Ru_1(x)) \rightarrow (Ru_1(t) \rightarrow Ru_1(x)). \qquad (9)$$

Statement (9) treats ORDER as an object whose properties are its source, its destination, and its contents. It says that when an order is received the contents of the order are carried out.

Statement (9) is a substitution instance of the more general form

$$\text{ORDER (originator, higher command) (recipient, xxxl system)}$$
$$\text{(stating, } \lambda) \rightarrow \lambda \qquad (10)$$

where λ is a general variable[*] whose substitution rules are part of the formation rules of the formalism and are not stated explicitly as part of the problem.

---

[*] We have been using Greek letters for such variables.

The semicolon is also used for introducing abbreviations that limit rules to particular problems. In (5) and (6) we used unbounded variables so that those rules would be general. If, for a specific problem, we wish to limit the rule to z aircraft, we can do this by restricting the subscripts of A to range from 1 through z in the conjunction of (5) and (6). If we have defined this conjunction as Ru_2 and have called our problem "problem 1" we would accomplish this by writing:

$$(\text{problem 1})\ Ru_2\ (i, j) \longleftrightarrow j \leq z \cdot \tag{11}$$

The machinery described up to this point is adequate for discussing solutions of resource assignment problems that are transformations in time and space, and do not involve very complex resource substitutions. Place and time are perhaps the simplest, and at the same time the most important, characteristics to be considered. The spatial and temporal coordinates of an object can be specified by a set of four-dimensional vectors. Because space and time are continuous, descriptions in terms of coordinates would involve an infinite set of such vectors. Rather than treating such sets as the lines (or arcs) they are, we shall treat them explicitly as functions of the characteristics of an object and its coordinates. For example, the speed of an aircraft would be treated as a function of its altitude and fuel load. Many of the characteristics of space and time as a set of points can be left implicit since they are implicit in the usual structure of the quadruples of real numbers.

RESOURCE-PLACE-TIME SUBMODEL

The formalism described above is somewhat unwieldy. The sample problem was a simple one, but stating it involved a large number of formulas, most of which were not easy to read. The complex notation was necessary for two reasons. First, there are a number of things usually taken for granted in problem solving that we have been making explicit. Also, we have been using abbreviations (mathematical symbols) not only to save space but also to make the structure of our statements more apparent to people familiar with this kind of symbolism. Both of these features are advantageous for setting up problems to be solved by computers, but they are awkward for people to use.

For this reason, in some applications we shall want to discuss only some aspects of the model, leaving the rest implicit. (Such subterfuges are familiar in mathematical logic.) To demonstrate such applications we shall discuss a problem from a 473L exercise.[*] We have been describing resources as bundles of properties. However, in most applications of the model to the 473L system, time and location are the most important properties. (The 473L system is largely concerned with problems having to do with moving resources from one place to another.)

From the RA model we can extract a submodel we have called the resource-place-time (RPT) submodel. In this submodel the only characteristic of resources considered is their location at various times. Such a submodel allows one to talk about all the parameters of routes and schedules without allowing formally for resource substitution; this aspect can be handled informally.

Plans call for an airlift to be mounted from X at hour H. Mounting the airlift requires getting a group of C124's and an air battle group to X by H-2. The sample problem arises when this schedule cannot be met because the C130's carrying the air battle group and the C124's clog the facilities required to get them to X. The submodel will examine this problem and look for all possible solutions. Some of these will be ruled out by intuition. Others will be reasonable possibilities which intuition alone might have overlooked.

The basic elements in our formalization will be states and missions. A <u>state</u> is an ordered triple of the form:

$$((\text{Resource}), (\text{Place}), (\text{Time}))$$

A set of such triples where the third element of the triple is the same (time) for all members of the set will be called a <u>time state</u>, or a <u>T-state</u>; <u>P-states</u> and <u>R-states</u> are defined similarly. A mission is an n-tuple whose first member is the resource assigned and whose other members are positive integers indicating which resource characteristics are believed to be essential to the performance of the mission.

---

[*] BANKO I. This example was discussed in our Concept Paper Number 7, from which much of this section of this appendix was drawn.

We recall the notion of an initial state and of a final state and hypothesize a problem with the geography shown in Figure B-11.
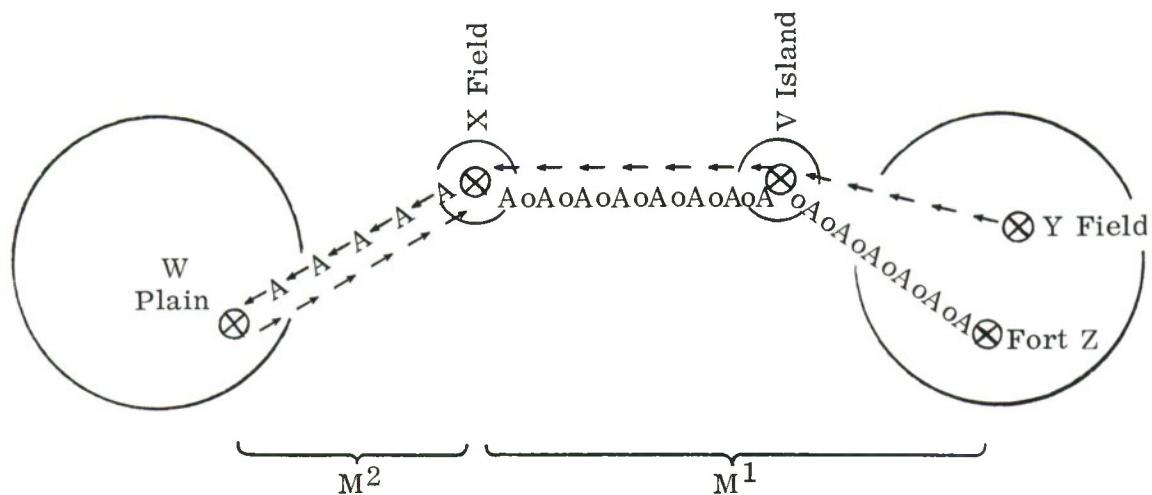
Key

A A A = Route of the ABG

→ → → = Route of the C130's

o o o = Route of the C124's

Figure B-11. Hypothetical Problem

The initial state in this problem is the three states:

$$\left.\begin{array}{l} (30 \text{ C130's}, \text{ Y}, 1 \text{ May}) \\ (1 \text{ ABG}, \text{ Z}, 1 \text{ May}) \\ (30 \text{ C124's}, \text{ Z}, 1 \text{ May}) \end{array}\right\} \text{ (P)}$$

The final state is:

$$\left.\begin{array}{l} (1 \text{ ABG}, \text{ W}, 3 \text{ May}) \\ (30 \text{ C130's}, \text{ X}, 4 \text{ May}) \end{array}\right\} \text{ (P)}$$

However, the system has several required intermediate states:

1.   The R-states that get the ABG to X field on the C124's

2.   The R-states that get the C130's to X field.

The system's work stops when this state of C130's and the ABG in X field are reached. Getting to this pair of intermediate states will be referred to as "Mission 1" or as "M^1."

The transition from the initial state to the final states requires a more detailed specification of the intermediate transition states. These transition states were arrived at with increasing specificity during the course of the exercise. The first step was to fill in the place values for the transition states. Such a set of transitions is called a route. For the C124's the route is:

$$
\left.
\begin{array}{l}
((\text{C124, loaded with 1 ABG}),\ (\text{Y Field}),\ (\ldots\ldots)) \\[1em]
((\text{C124, loaded with 1 ABG}),\ (\text{V Island}),\ (\ldots\ldots)) \\[1em]
((\text{C124, loaded with 1 ABG}),\ (\text{X Field}),\ (\ldots\ldots))
\end{array}
\right\} \quad \text{Route } ^{\text{C124}}
$$

A route with the blanks in the time spot specified is called a schedule. The problem with which we are concerned requires a schedule, with its final state meeting the conditions of the problem (P). Let us assume that the C124's arrive at W on 4 May.

We postulate 1:00 on 1 May as H-hour and number our hours from it. Since aircraft cannot all land and take off from a base at the same moment, our notion of a schedule must be extended to allow us to represent the time periods during which bases are used. For this purpose, we might write down the beginnings and ends of field-use states as two states. We abbreviate by writing "((a), (b), (c-d))" for "((a), (b), (c)), ((a), (b), (d))." Thus, we might have the following situation:

1.   Mission 1 as ordered:

Initial state (S^I):

$$
\left\{
\begin{array}{l}
((30\ \text{C130}),\ (\text{Y}),\ (1{:}00)) \\[1em]
((30\ \text{C124}),\ (\text{Z}),\ (1{:}00)) \\[1em]
((1\ \text{ABG}),\ (\text{Z}),\ (1{:}00))
\end{array}
\right\}
$$

Final state (S^F):

$$\rightarrow \left\{ \begin{array}{l} ((30 \ C130), \ (X), \ (61{:}00)) \\ ((30 \ C124), \ (X), \ (61{:}00)) \\ ((1 \ ABG), \ (X), \ (61{:}00)) \end{array} \right\}$$

2.   Mission 2 as ordered:

Initial state:

$$\left\{ \begin{array}{l} ((30 \ C130), \ (X), \ (63{:}00)) \\ ((1 \ ABG), \ (X), \ (63{:}00)) \end{array} \right\}$$

Final state:

$$\rightarrow \left\{ \begin{array}{l} ((30 \ C130), \ (X), \ (100{:}00)) \\ ((1 \ ABG), \ (W), \ (90{:}00)) \end{array} \right\}$$

The arrows above point to those parts of missions that are crucial to the next mission. (Note that we may not always know where to write all the arrows and that the arrows do not tell us what it is about the state that is critical.)

We also have a schedule, or set of schedules, one for each of the resources involved in the missions:

3.   Schedules for the execution of M^1:

a.   Schedule for C124's:

$$S^{C124} \left\{ \begin{array}{l} ((30 \ C124's), \ (Z), \ (1{:}00 - 10{:}00)) \\ ((30 \ C124's), \ (V), \ (25{:}00 - 45{:}00)) \\ ((30 \ C124's), \ (X), \ (55{:}00 - 61{:}00)) \end{array} \right\}$$

b.   Schedule for the ABG:

$$S^{ABG} \left\{ \text{(Same as 3a. The ABG is loaded onto the C124's.)} \right\}$$

c. Schedule for the C130's:

$$
S^{C130} \left\{ \begin{array}{l} ((30 \ C130\text{'s}), \ (Y), \ (10{:}00 - 20{:}00)) \\[1ex] ((30 \ C130\text{'s}), \ (V), \ (45{:}00 - 60{:}00)) \\[1ex] ((30 \ C130\text{'s}), \ (X), \ (75{:}00 - 81{:}00)) \end{array} \right\}
$$

The statements of the missions serve to provide reference points for the examination of the schedules, which are the "solution" of the Air Force to the problem stated in the mission. In this case, the scheduling task[*] has produced an inadequate solution to the problem. We require a problem-solving algorithm, operating on the sets of symbols written above, to transform the final state so that it is in closer agreement with the mission.

Consider the algorithm suggested by the following two rules:

1. Every step in the algorithm consists of attempting to alter an element in the schedule in the direction suggested by the nature of the discrepancy to be altered.

2. The elements in the schedule are to be examined from right to left, and then from bottom to top.

The schedule as it stands is somewhat incomplete. Therefore, we include at the end of the schedule the appropriate final state of the mission statement, and at the beginning of the schedule the appropriate initial state of the mission statement. This structure and the order of search (with some of the solutions indicated) are shown in Figure B-12.

In this illustration the schedule is represented schematically as a series of P-states organized into a schedule. A line indicates the order in which the elements of this schedule are examined for a solution. Only solutions derived from an examination of the time elements are indicated. (The part of the arrow up the R-elements does not make much sense in this RPT submodel, since the R-elements are all the same. However, its purpose will be clearer when we discuss further characteristics of the full model.) The solutions are defined on p. B-28f.

---

[*] "Task" is defined on p. B-11.
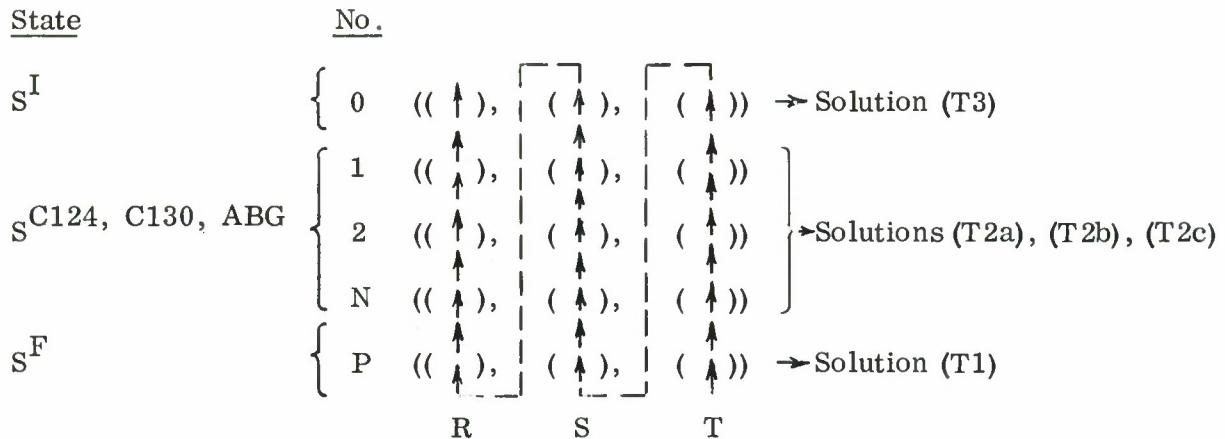
Figure B-12. Order of Examination

We shall begin to carry this algorithm through to show how solutions might arise from such a process. We begin with the first column. As we proceed we shall find that our version of the schedule is incomplete; therefore we shall add elements as they appear to be needed.

First we consider the possibility of allowing the final T-state to stay as stated in the schedule. That is, we compare T_N of the schedule with the desired state T_P and, noticing a discrepancy, consider the possibility of letting this discrepancy stay as it is. Another way of looking at the situation is that we attempt to adjust T_P to minimize the difference expressed by $T_P < T_N$. This results in solution T1:

Call the appropriate authority and suggest slipping the arrival time.   (T1)

Since the exercise being discussed was a normative one, each possible solution that was inappropriate was ruled out by some situation in the world. In this case, incoming messages had indicated the slippage proposed in (T1) to be highly undesirable. Also, it had been indicated to the system that slipping was inappropriate unless all other alternatives had been exhausted.

Next we consider the set of times that indicate when planes are to leave and arrive at the various points on the route. As a result of this investigation, we might propose to the scheduling command:

> Can the schedule be tightened up, e.g., can planes be more closely spaced in the schedule? (T2a)

> Can the speed of individual planes be increased? (T2b)

> Can the amount of time on the ground be decreased, i.e., speed up refueling? (T2c)

Proposal (T2c) includes two possibilities: Either speed up the ground crews (decrease time on ground to X (X > 0)) or try aerial refueling (X = 0).

Although our formalism has suggested solutions to the problem, these solutions have not resulted from formal manipulations. Our intuition of the nature of the situation was used in producing the solutions. This is the basic difference between a formalism and an intuitive description of it.

The notion of a state can be extended in a natural way if we observe that our current notion consists of a series of snapshots. Since the problem here is one of time, we might generalize to the notion of movies (maintaining our metaphor). We thus consider what we might call "steady state" movies, where steady states are the only things depicted. Thus we might have the following:

	R	P	T	Steady State
	(C130)	(Y)	(10:00 - 20:00)	Loading troops
	(C130)	(Y)	(19:00 - 26:00)	Taking off
	(C130)	(10 Kft.)	(20:00 - 49:00)	Flying
S^{C130}	(C130)	(V)	(45:00 - 50:00)	Landing
	(C130)	(V)	(46:00 - 58:00)	Refueling
	(C130)	(V)	(50:00 - 60:00)	Taking off
	(C130)	(15 Kft.)	(51:00 - 80:00)	Flying
	(C130)	(X)	(75:00 - 81:00)	Landing

If we decompose this schedule into the schedules for individual planes in order to overcome the overlap of T-states, labeling the first member of the T-element in the extended version as $I(T_i)$ and the second as $E(T_i)$ for each state i,[*] we can now define the dependence of the final state on the others (in the T dimension) as follows:

$$E(T_N) = T_{(N-k)} + \left\{ E(T_{N-k}) - I(T_{N-k}) \right\} + \ldots + \left\{ E(T_N) - I(T_N) \right\}$$

The states of S^{C130} can then be defined in terms of the states of its elements. Since the problem is to decrease $E(T_N)$, at least one way of accomplishing this is to decrease the size of the terms of the above equations. This gives us T2a, T2b, and T2c in a fairly formal way.

If we add the further restriction that all times within a mission are defined by the indicated function, while no times outside are thus defined by the system being exercised, we see that any attempt to change the values for $I(T_1)$ or for $E(T_N)$ must come from outside. Thus we get the last two possibilities: Ask for a slip in time or ask for the planes at an earlier initiation time. The one we have not mentioned so far is:

Can the whole operation be initiated a day earlier? (T3)

This appears (intuitively) to exhaust the possibilities in the time dimension.

Now we return to the final state of the P-component in the state vectors as indicated in Figure B-12. Actually, there are three final states: the final state of the ABG, the final state of the C130's, and the final state of the C124's. The ordering of the P-states depends, in some sense, on the shape and nature of the surface of the earth, while the ordering of the T-states is the straightforward ordering of "< ." Therefore, we postulate the following mode of "slipping the final state in the P-dimension." To slip a state P_i is to substitute P_{i-j} for it, where $j < 1$. Slippage

---

[*]We assume that we have indexed the states with ascending positive integers.

here is the possibility of leaving something at an earlier location. This suggests the following solutions, derived analogously to T1 - T3:

Leave the C130's in Z or V	(P1a) and (P2a)
Leave the ABG in Y or V	(P1b) and (P2b)
Leave the C124's in Y or V	(P1c) and (P2c)
Forget about the C130's	(P3a)
Forget about the ABG	(P3b)
Forget about the C124's	(P3c)

The P3's require a slight extension of our notion of the ordering of P-elements.

Slippage in the P-dimension in this kind of problem implies a degradation in the final state, which might be made up in some manner. (A similar situation might be said to occur in the slippage of time. If the authority of the system being exercised were somewhat different, it might attempt to minimize the effect of time slippage (i.e., late initiation of M^2) by a bombing attack or some other supporting action.) Therefore, it is important to consider a means of minimizing the effect of the slippage (P1 - P3). Minimization will depend on the nature of the resource being left behind. In the case of P1a, P2a, and P3a, leaving the C130's in either Z or V seems to have about the same effect. This need not be the case. (Suppose the C130's had some sort of reserve function such that delivering C130's in V allowed the release of other C130's already at X. This fact suggests that the importance of location (P) might be indicated in the mission vector.)

The mission of the C130's is to airdrop troops. Their loss can be minimized in any of the following ways:

(1) Have the mission performed by other aircraft already in the theater.

(2) Have the mission performed by some non-aircraft (e.g., ships, trucks).

(3) Have the mission performed by a resource in some other part of the mission (in this case the C124's).

Solution (1) could include (3) since the C124's will be in the area at the appropriate time, but this point has been mentioned explicitly since their presence at that time is likely to be overlooked. However, (1) might be generalized to:

> (1') Have the mission performed by some aircraft that will be in
> the theater by T_N.

Solution (1) is assumed ruled out by the fact that the area command would not request C130's if it had something else to use. Solution (2) is ruled out by time and considerations of location, and (3) is ruled out (in a manner we have not yet specified) by the characteristics of the C124's as resources.

A more complex inference allows one to consider separately the leaving of the aircraft at Z and at V (or even leaving them in midair). The argument might be "Does leaving C130's in V produce any other resource we could use instead (thus minimizing the effect of the slippage)?" There might be C130's at X (being held there to support some mission between X and V) which could be released if aircraft were available at V. Leaving the C130's at Z might be ruled out because it was assumed to have been considered when the use of C130's from Z was selected. Nevertheless, this gives us a fourth alternative:

> (4) Have the mission performed by some other resource at V
> (or X) at time $T_V(T_X)$.

We will not discuss the applications of this algorithm to the C124's or the ABG except to point out that P1c (3) was the "correct" solution to the problem. The fact that the correct solution results from this algorithm is not of interest since we knew that it would. The fact that it results late in the algorithm suggests the use of heuristics based on characteristics of individual problems in order to improve the efficiency of the algorithm.

It might be worthwhile to validate this submodel by investigating whether the people who work in an actual system would suggest any solutions not produced by this algorithm. The structures presented here could be used to characterize the steps they actually go through. There are undoubtedly certain alternatives which

never occur to personnel in a real system on the ground that they are totally inappropriate system behavior. (One such example is T1a.)

Given a solution, it is not always possible to determine what sort of solution it is, e.g., to say that it is a solution of the form T1a. The algorithm is redundant in that it produces the same solutions at several points.

The set of P-solutions considered above has not included the effects of transformations of the in-flight P-states. Such transformations are difficult to handle in a formal way, but the possibility should be mentioned here since the correct solution could be the result of such a transformation. (Assume that by changing altitude the aircraft could ride (or avoid riding) the jet-stream and thus increase its ground speed.) The difficulty in formalizing what is involved in this type of step is that there are so many transformations of the interim stage that it would be difficult to list them, even for so unimaginative an algorithm as an exhaustive search. This suggests that the formalization we have considered remain partial and be augmented by intuition.

THE FULL MODEL

So far we have concerned ourselves only with physical objects and only with one kind of relationship between them. We have considered spatio-temporal positions as characteristics of objects and described rules for transforming these characteristics. The characteristics themselves have been considered as a set of points in a four-dimensional space (Figure B-13).

Other characteristics of objects can be treated analogously. In the rest of this section we shall indicate some differences which will arise. The differences will fall into three categories: (1) the sets of points to which properties are referred will have different characteristics, (2) the objects will be different, and (3) the kinds of transformations required will need to be handled by somewhat different machinery.
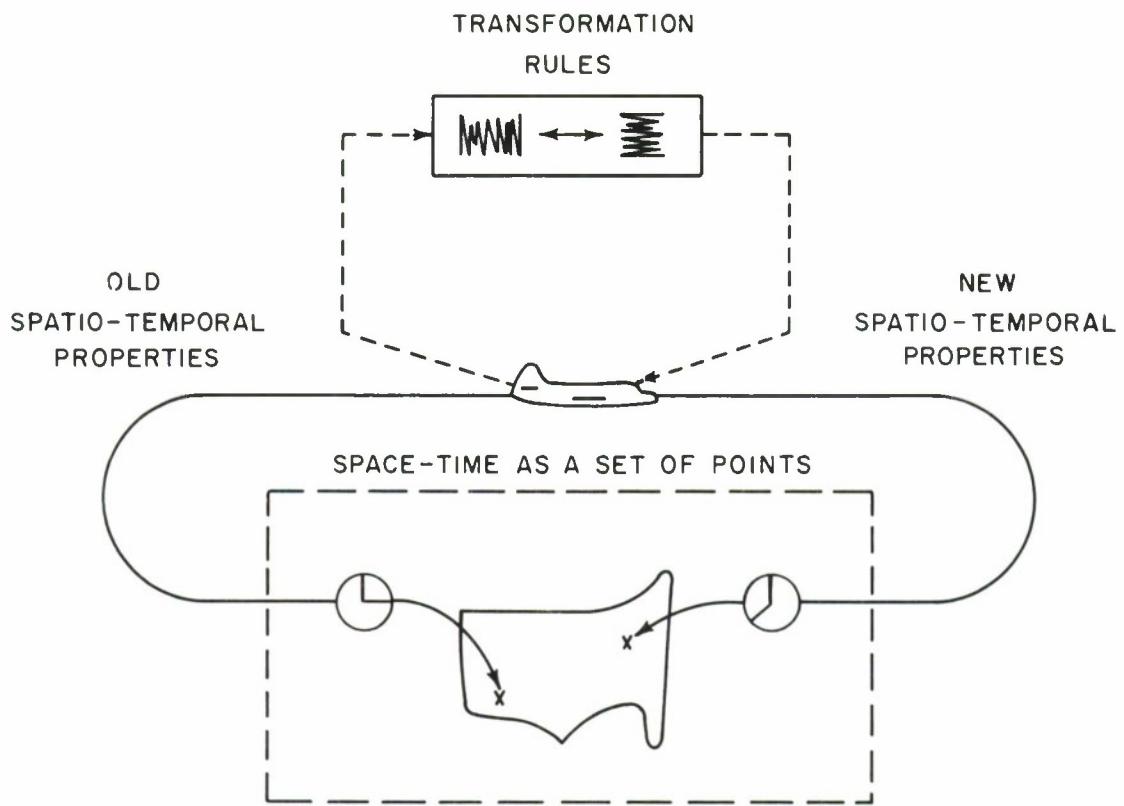
Figure B-13. A Space as a Set of Points and a Set of Transformation Rules

Before discussing the changes in (3) we can briefly list the varieties possible in (1) and (2):

(1) Possibilities in the space as a set of points:

   a. The set of points is continuous and isomorphic to the real numbers (e.g., rate of climb, fuel consumption). Characteristics can be described as analytic functions, and much of the machinery of traditional mathematics can be used.

   b. The set of points is discrete and isomorphic to the integers (e.g., number of troops, fuel capacity). The machinery used is that of arithmetic, of the algebra of rings (for the first example), and of fields (for the second example).

c. The set of points is discrete but lacks the natural metric of the integers or is not linearly ordered (e.g., command structures, utilities associated with plans). The machinery used is the weaker but more general parts of algebra such as lattice theory.

d. The set of points is monoid (or free semi-group) generated from some finite alphabet, usually A, B,...Z, 0, 1,...9 (e.g., tail numbers of aircraft, names of airfields). The machinery used includes fragments of algebra and mathematical logic.

(2) Various types of objects

We shall be concerned with two types of objects: physical and abstract. The abstract objects will fall into three subcategories: classificatory, command, and intentions. The class C130 is a classificatory abstract object although any particular C130 is a concrete object. A squadron is a command abstract object, and a mission is an intention. There are various relationships between such objects; physical objects can be parts of each other or can have the same locations. They can belong to classificatory objects, be assigned to intentional objects, and so forth.

(3) Transformations

The basic relationship will be denoted by ϕ. Roughly, $\lambda\phi\gamma$ means that λ is a part of γ, but how this is to be interpreted depends on the nature of the objects being related. For concrete objects,

$$\text{engine} \quad \phi \quad \text{aircraft} \tag{12}$$

means that the engine would be "a part of" the aircraft. Here, the removal of "engine" changes the characteristics of the aircraft. To say that an aircraft is part of a command object is a similar statement. Thus

$$\text{aircraft} \quad \phi \quad \text{squadron} \tag{13}$$

means something similar to (12). One difference is that the rules governing the changing from one command to another are quite different from those governing the changing of parts of physical objects. Changing engines may require a mechanic and a certain amount of time. Switching an aircraft from one squadron to another may involve only a piece of paper. A still different situation occurs when one relates a physical object to a classificatory one as in

$$\text{aircraft} \quad \phi \quad \text{C130.} \tag{14}$$

Here the removal of the aircraft from the class is a virtual impossibility. The aircraft may be destroyed, but it remains a destroyed C130; the class of C130's is not changed since C130's continue to have the same characteristics. The only change is in the number of objects fitting into this classification.

The sense in which an engine is part of an aircraft is different from the sense in which an aircraft is part of the airfield at which it lands; it is also different from the way in which a mission is part of a plan. One can distinguish between (12) and the case of an aircraft which lands at an airfield by introducing an abstract object (the precise location) of which both the aircraft and airfield are parts, while the ϕ relationship of (12) applies between physical objects.

Let us define additional operators Θ_1 and Θ_2 such that:

$$\Theta_1(\text{xxx})(\text{YYY}) = \text{zzz}$$

$$\text{and} \quad \Theta_2(\text{zzz})(\text{YYY}) = \text{xxx} \tag{15}$$

$$\text{if and only if } \text{YYY} \ldots (\text{xxx, zzz}) \ldots.$$

This additional notation allows one to talk about two things in the same place. If we have

$$\text{LA GUARDIA AIRPORT (location}^1\text{, new york city)} \tag{16}$$

and

$$\text{EMPIRE STATE BUILDING (location}^1, \text{ new york city)} \qquad (17)$$

we have:

$$\Theta_1 \text{(location)}^1 \text{(LA GUARDIA AIRPORT)} = \Theta_1 \text{(location)}^1 \text{(EMPIRE STATE BUILDING)} \quad (18)$$

which says that La Guardia and the Empire State Building are at the same place. One needs a more precise indication of location (location^2) to be able to say that an aircraft landing at an airfield is in the same location as the airfield. From the above, an aircraft crashing into the Empire State Building would be said to be at La Guardia. The detail with which one specifies locations and other abstract characteristics will depend on the purpose to which one wishes to put the model.

Most other distinctions can be handled similarly by introducing specific abstract objects so that ϕ remains a relationship that corresponds to the intuitive notion of "part of." This simplifies a number of matters. This simplifies the statement of transformation rules defined for large classes of objects in terms of what is required to make and break these relations and what occurs when they are made and broken.

Another advantage of the use of a single relation is that much of the reasoning required to solve resource assignment problems is based on the fact that ϕ is somewhat transitive. Thus, if an aircraft lands at LA GUARDIA,

$$\Theta_1 \text{ (location}^2)(\text{A/C}) = \text{ LA GUARDIA AIRPORT} \qquad (19)$$

and

$$\Theta_1 \text{ (location}^1)(\text{LA GUARDIA AIRPORT}) = \text{ NEW YORK} \qquad (20)$$

we can deduce that, to the degree of fineness expressed by location,^{1} the aircraft is in New York. The same sort of transitivity holds in general. Thus, from

$$\text{aircraft } \phi \text{ C130} \qquad (21)$$

and

$$C130 \quad \phi \quad transport \tag{22}$$

one can derive

$$aircraft \quad \phi \quad transport. \tag{23}$$

This suggests an extension of the definition of Θ_1 so that the transitivity-inherited properties of the aircraft (those which it has because it is a transport) are ascribed to it. Thus we can redefine Θ_1 as Θ_1':

$$\Theta_1' \, (xxx)(YYY) = zzz \text{ if and only if there exists a } YYY(xxx, zzz)$$
$$\text{or a } YYY \, \phi \, y'y'y' \text{ and a } y'y'y' \, (xxx, zzz). \tag{24}$$

A Θ_2' can be defined similarly.

The relationship ϕ and the operators Θ_i' appear to provide the medium for a powerful model for problem solving. However, since the model will tend to be better suited for computers than for people, the limited kinds of models illustrated by the RPT submodel discussed above may be preferred for many applications.

RECOMMENDATIONS

The resource assignment model appears to provide a medium for formalizing the processes involved in problem solving in certain kinds of command and control systems. An effort to apply this model to an actual problem still remains to be made. Such an application is probably best done on a computer. The utility of such a model for application to actual problem solving will require the use of heuristics, and different schemes for such heuristics must be investigated. Finally, the utility of intuitive submodels (illustrated by the resource-place-time submodel) to the exercise and evaluation of a command and control system might be validated in practice.

REFERENCES

1. A. Newell, J. C. Shaw, and H. A. Simon, "Report on a General Problem-Solving Program," <u>Proc. Internatl. Conf. on Information Processing</u> (Paris, France: UNESCO House, 1959).

2. C. E. Shannon, "A Mathematical Theory of Communication," Bell System Tech. J. <u>27</u> (1948).

3. G. Frege, "Begriffsschrift" (Halle, 1879).

4. W. O. Quine, "Methods of Logic" (New York, N. Y.: Henry Holt and Company, Inc., 1950), pp. 220-224.

5. S. C. Kleene, "Introduction to Metamathematics " (New York, N. Y.: D. Van Nostrand Company, Inc., 1952).

6. J. H. Proctor, "Normative Exercising: An Analytical and Evaluative Aid in System Design," paper delivered at the First Congress on the Information System Sciences, Session XV, Hot Springs, Virginia (20 November 1962).

APPENDIX C

THE EXPECTED UTILITY MODEL

APPENDIX C

THE EXPECTED UTILITY MODEL

The expected utility (EU) model relates a command and control system's inputs and its "character" to its behavior. The model is closely related to von Neumann and Morgenstern's[1] notion of a "game in extensive form."

The model can be applied in two stages. The first stage yields a characterization of a system's strategy from observation of its behavior during an exercise. The outputs of this stage provide part of the inputs for the second stage of application. The second stage of application yields a prediction of system behavior in a rather general form. Such a generalized prediction is intended to be employed by system users to determine whether the predicted behavior is what they want.

This appendix is divided into four parts. The first part discusses the applications of the model, the second part discusses the inputs and outputs of the model, and the third part describes the elements of which EU models of particular systems can be constructed. The appendix concludes with a brief summary and some suggestions for further work which might be performed before the model is applied to a particular command and control system.

PURPOSES

The expected utility model applies the notion of a game in extensive form to command and control systems in an effort to both characterize and to predict their behavior, and it is intended to contribute to exercise construction, control, and evaluation. The EU model is a partial effort to represent the decision-making part of command and control system activity. Since this is predominantly a human activity, this model may provide a useful tool for evaluating the characteristics of the human elements in a command and control system. In particular, it may eventually prove useful for determining the uniformity or stability of system performance with changes of personnel. It also seeks to give a precise description of activity which the rest of a command and control system's work is intended to support. Thus it may help to provide criteria for evaluating the system's work and the machinery with which that work is done.

The EU model assumes that an exercise is punctuated with decision points, each of which requires the system (or some of its personnel) to choose one from among several possible alternatives. The model attempts to describe what such choices involve insofar as they are "rational." Rationality is defined (roughly) as the choice of that action whose apparent utility toward attaining system goals is maximal, assuming only the given information.

The EU model has been described by Ward Edwards[2] in connection with the prediction of choices among bets. Edwards assumed that the utility of payoffs was well-defined and additive, because he equated monetary and utility values. He also assumed well-defined and easily-determined probabilities. However, in using this model to describe decisions in a command and control system, one cannot make these assumptions because the values of probabilities and utilities are fairly elusive in this area.

If one can devise a technique for obtaining a reliable[*] measure of the utilities and probabilities involved in system choices, the EU model may provide a basis for experimentally determining the strategies[†] of a given system. Once one has run exercises to determine system strategies, these results might be used to predict system performance for a large class of problems. Such predictions in turn can provide the basis for an evaluation of system personnel and for changes in the ways in which the basis for decisions (information, guidelines, and the like) are given to personnel.

INPUTS

As indicated above, the EU model can be applied in two stages. The first of these is an exercise to determine the nature of a given system's strategies (Figure C-1). The inputs to this application are the utility and probability structure of

---

[*] A measure is said to be "reliable" if it gives comparable results each time it is applied to the same thing, even when this application is made by different people.

[†] A "strategy" is the way in which a given system interprets a given probability/utility structure and/or the ways it acts in the face of problems having such structures.

a problem presented to a system in an exercise and the system's response to this problem during the exercise. The output is a "profile" of system strategy (Does it tend to take great risks or avoid them; is it very sensitive?). The second application takes, as its inputs, the utility/probability characterization of either a single
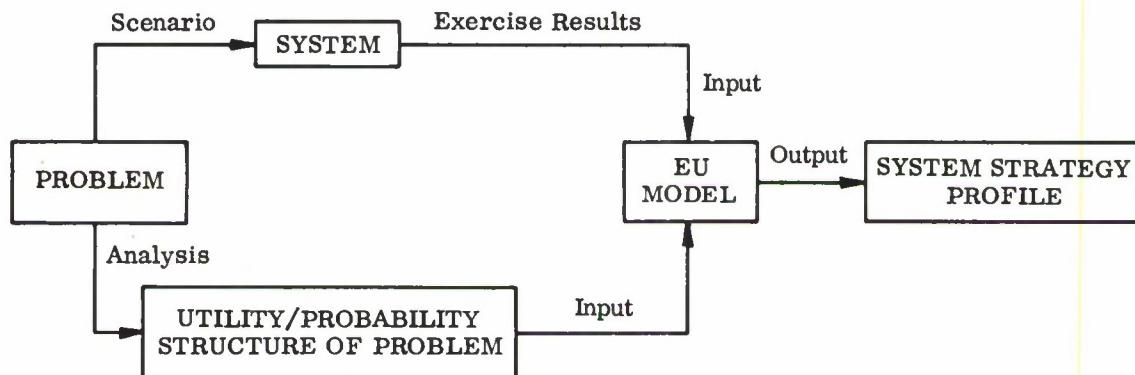
Figure C-1. Stage 1 in the Use of the EU Model

problem or of a class of problems together with the (empirically determined) profile of a system's strategy, and produces a prediction of system response to these problems (Figure C-2). It should be clear from this description that the EU model is more properly a medium for evaluating a system as a whole (or at least its personnel and procedures) than for evaluating individual system personnel.
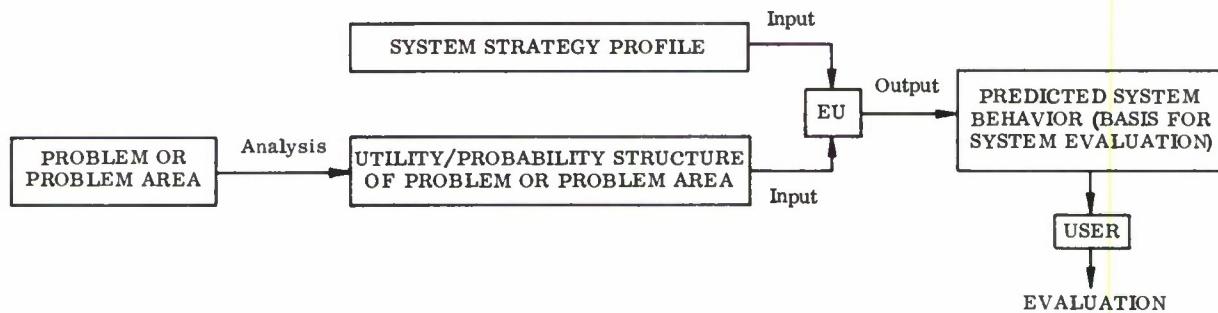
Figure C-2. Stage 2 in the Use of the EU Model

In order to apply the model, one must have ways of measuring the values of the independent variables (probabilities and utilities). Devising these is no trivial task; one would probably have to be satisfied by fairly rough approximations. Rating by qualified judges (appropriately instructed) might be used to provide a scale. Caution must be observed here, however, in that even qualified judges need not be aware of the probabilities or utilities upon which they would act, let alone anything approaching inter-subjective probabilities or utilities. Some of the difficulties here have been discussed by Edwards.[3]

OUTPUTS

Assume that the probabilities and utilities assigned in a problem characterization are well-ordered* and additive. At each decision point (i) there will be (b_i) alternatives (Figure C-5, p. C-12) $A_1, \ldots, A_{b_i}$. Let $O_1^i, \ldots, O_j^i$ be the (j) outcomes of selecting A_i, let P_k^i be the probability of O_k^i, and let U_k^i be the utility of this outcome. The system which selects the alternative A_m which maximizes

$$\sum_1^i P_i^m \cdot U_i^m$$

might be considered an ideal[1] with which the observed behavior of a system can be compared.

However, in practice, neither complete additivity nor well-ordering can be assumed. In general, one has only a partial ordering.† This can sometimes be turned into a well-ordering by using data from experiments to force a system to choose between members of "unconnected" sets. The results of such experiments might then be used to rank the members of these previously unconnected sets. In general, when such results are obtained one cannot assume that the ordering imposed will necessarily be consistent with the original partial ordering. Lack of

---

* A set is said to be "well-ordered" if every subset has a first element by the ordering relation.

† A partial ordering is a union of well-ordered classes.

additivity might be eliminated by correction functions $f(U_i)$ applied to the original utilities U_i so that the $f(U_i)$ are additive, even where the U_i are not.[*] For example, a particular system might seek to avoid outcomes of great negative utility (catastrophes) far more than it seeks outcomes of positive utility of the same magnitude. Such a strategy might be called an "avoidance of loss" strategy. If we introduce the notion of a "utility to a system" (as well as that of a utility per se), we might diagram such a strategy as in Figure C-3.
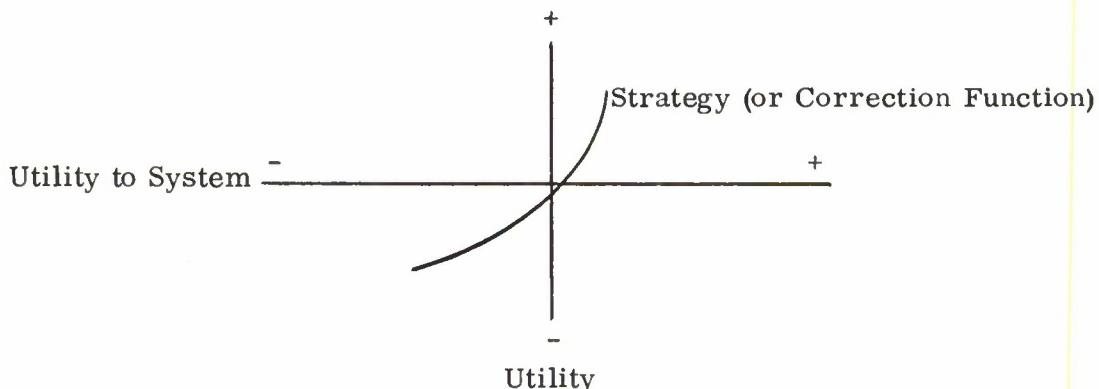
Figure C-3. Avoidance of Loss (Cowardly) Strategy

Another feature which might be included in the strategy profile is the degree of certainty required before a system makes a decision. One might extend the model to permit ranges of probabilities and utilities between bounds (possibly with various distributions between these bounds). During the running of an exercise, these ranges will tend to narrow as more and more information becomes available, or to widen as the situation becomes more complex or information more confused. In terms of such ranges, one might define a system's willingness to take on poorly defined situations, a different aspect of its strategy profile.

---

[*]Correction functions are equivalent to the "strategy profiles" mentioned on p. C-17. The difference is that the correction function is a correction applied to the values to be maximized, while a strategy profile changes the description of the system behavior.

In the second stage of application, these profiles are themselves inputs (together with the utility/probability structures of problems), and the outputs are predictions of system behavior. These can be compared with the desired behavior for evaluation purposes.

DESCRIPTION OF EU MODEL[*]

A "decision" is defined here as the conscious choice of one of several alternate courses of action. We shall describe a simple model for the steps involved in the making of such a decision. The model design rests on several assumptions:

1. The decision maker realizes that he must make a decision.

2. The alternatives between which the decision maker must choose are apparent to him.

3. The supportive data necessary to make a decision (e.g., characteristics of alternatives, amount of goal satisfaction provided by each alternative, consequence of choice of each alternative) are also available to the decision maker.

4. The choice of alternatives is made on a rational basis.

It will be apparent to the reader that these assumptions do not always apply in the real world. However, in the exercise and evaluation environment in which the exerciser controls the world directly and thus the system indirectly, it may be possible to force these assumptions.

The stage which precedes the decision process might be called the "presentation of alternatives," in which the decision maker is confronted with several courses of action (see Figure C-4). This confrontation will bring to the attention of the decision maker the necessity for making a decision, but the decision need not be forced upon him. It is possible for the decision maker to solve a problem without

---

[*]This section of this appendix is taken largely from our Concept Paper No. 6. Familiarity with the contents of this paper is not assumed.
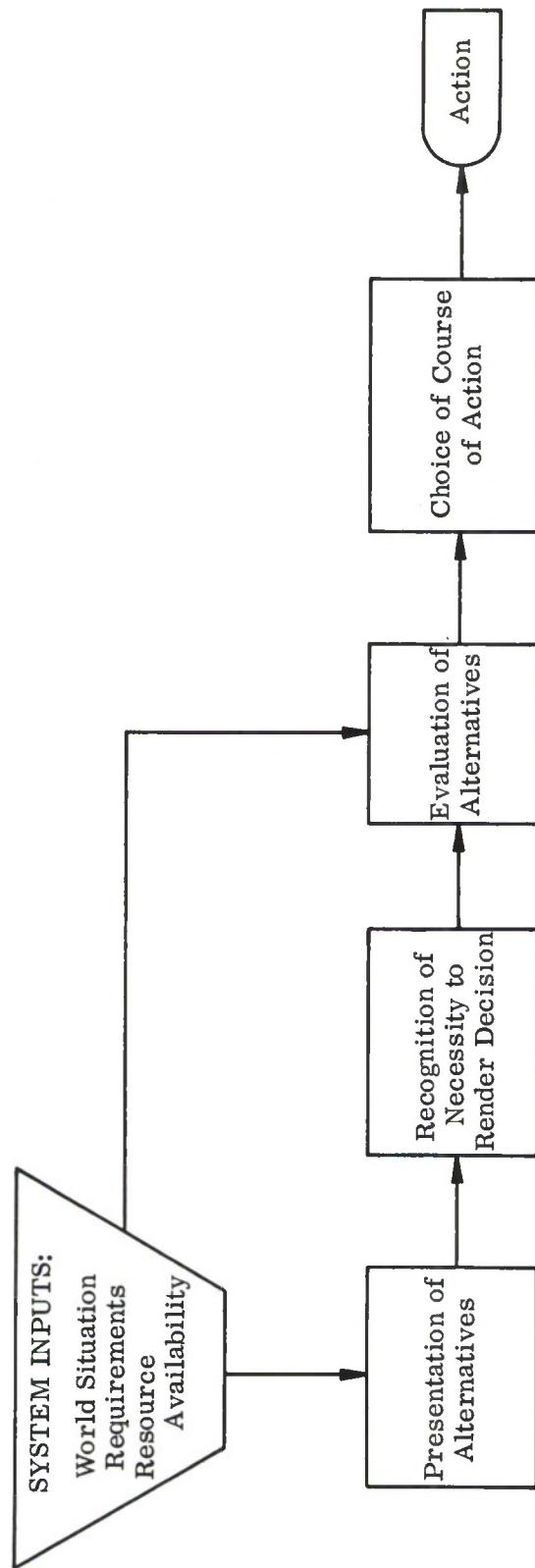
Figure C-4. Stages in Decision Making

SYSTEM INPUTS:

World Situation
Requirements
Resource
Availability

Presentation of
Alternatives

Recognition of
Necessity to
Render Decision

Evaluation of
Alternatives

Choice of Course
of Action

Action

ever actually making a decision, by proceeding along the only path apparent to him and remaining oblivious to points where alternate paths might have been chosen. This oblivion may result from ignorance or set, but in this paper we will deal with the decision maker who is aware of the choice points in the problem and makes a rational choice at each point where more than one alternative exists. We assume that the decision maker is part of a goal-directed system and that his actions will be directed toward maximum achievement of goals.

The decision process begins with an evaluation of the alternatives. Probably the most important consideration in this evaluation is how each alternative would advance the system toward its goals. In most cases this is not a straightforward process since the relationships between goals are rather complex. There are immediate, intermediate, and long-range goals (probably a continuum rather than step function); military, national, and international goals; local, theatre, command, and service goals. Trade-offs must be made when some goals are served positively, others negatively, and some not at all. We assume a hierarchic structure of goals which may vary from time to time during an exercise. It is up to the system exercisers to convey to the system information about that part of the goal structure which cannot be assumed on the basis of previous experience. The decision makers form their own concepts, based on the input data, and attempt to achieve those goals which appear paramount to them.

The evaluation of the alternatives makes use of supportive data such as plans, aircraft characteristics, and runway characteristics. These data may be contained within the memory of a person or persons, the memory of the computer, the pages of a book, the face of a chart, etc. Its form is not material to the model; rather, it is important that the data are available if the decision maker should choose to call on them.

How is the decision maker apprised of the necessity to make a decision? There are three states along a continuum of what might be called "system decision awareness." This is a scale beginning at relative unawareness and ending in complete cognizance of the decision and its elements.

The first state we call the "anticipation state." In this state, the system (or one of its members) concludes that a decision or choice among several alternatives

will be required because of the way the world about the system is behaving. An example of this state might be an occasion in an exercise when the system members must decide whether or not to summon superiors. If the situation deteriorates at the present pace, then these superiors will be needed. If it improves, the additional staff will not be required. The decision to call in the superior staff would most probably be made on the basis of a steadily worsening world situation and a prediction that this trend would continue. In the anticipation state, as in the "mission matching" to be described next, the roles which the system members visualize for the system and for themselves play a particularly important part in whether or not a decision is made. If the system member(s) does not foresee that system involvement will occur in a specific situation, then no system action can be expected.

The second system state is called the "mission matching" state. In this, the world situation appears to the system member(s) in a configuration which corresponds to one defined in the system's mission statement and requires a decision. For example, if the system's mission statement requires the system to select a scheduling agent for a multi-command operation as soon as the participators have ascertained the resources to be committed, the mission matching state exists when the system selects a command to do the scheduling for an airlift operation.

The third state is the "order forced" state in which an order or directive from a higher command echelon directs the system to render a decision. An example might be a response to an order from JCS which states: "By 0400 select the command to perform airlift and notify JCS."

A chart might be constructed for each phase of an exercise listing the messages required to move the system through each phase and the time at which each message is to be transmitted. Messages would be listed for each possible or desired state for each phase. In some instances it may not be possible or desirable to plan ahead for the traversal of all three states, in which case messages for only the selected states would be prepared. A skeletal example of a message table is shown in Table C-1.

TABLE C-1

MESSAGE TABLE

Time	Anticipation State (Message No.)	Mission Matching State (Message No.)	Order Forced State (Message No.)
t_1	1		
t_2	2		
t_3		3	
t_4		4	
t_5			5
Checkpoint 1			
t_6	6		
t_7		7	
$t_8 \cdot$			8
Checkpoint 2			
t_9		9	
t_{10}		10	
t_{11}		11	
t_{12}			12
Checkpoint 3			

Because the time of transmission of each message is known to the exerciser and because the time of occurrence of a checkpoint (after the terminology of Proctor[4]) is an observable event, it will be possible to obtain quantitative time measurements for system phase traversal. Thus it will be possible to obtain a

measure of system sensitivity—not only for the system as a whole but also for its components (since various components will be primarily concerned with different portions of the exercise). Areas in which the system is found to be deficient may be selected for special training.

The exerciser's manipulation of this process is somewhat complicated by the fact that men as well as machines are being manipulated. If we know how a machine is programmed, we can predict with certainty what its response to a specific stimulus will be. We cannot be quite so certain about the people in the system. Set, emotions, incomplete knowledge, superior insight, and a host of other factors lead to decisions made not wholly on a rational* basis. However, the individual will not always optimize the system's situation. Garfinkel[5] has described von Neumann's rational game player thus: "He never overlooks a message; he extracts from a message all the information it bears; he names things properly and in proper time; he never forgets; he stores and recalls without distortion; he never acts on principle, but only on the basis of an assessment of the consequence of a line of conduct for the problem of maximizing the chances of achieving the effect he seeks." One might easily question whether any decision maker constructs a utility table, evaluates this against his risk philosophy, determines the probability of each possible decision outcome, and evolves a logical decision unaffected by group dynamics. Yet one can assume such a utilitarian decision maker as an ideal. This assumption leads us to a useful and manageable model which, in spite of its abstraction, provides a standard against which to measure actual decision makers.

In our model, a decision is the result of an evaluation of two factors:

1. The utility of each alternative action

2. The probability of achieving each of the possible outcomes by choosing any one of the alternatives.

Figure C-5 presents a graphic expression of this concept.

----

*In order to formulate what is meant by a strictly rational decision in a given situation, certain elements must be carefully defined, such as the probability distribution of all possible contingencies, the class of available resources, and the set of values, goals, and objectives. This may not always be possible.
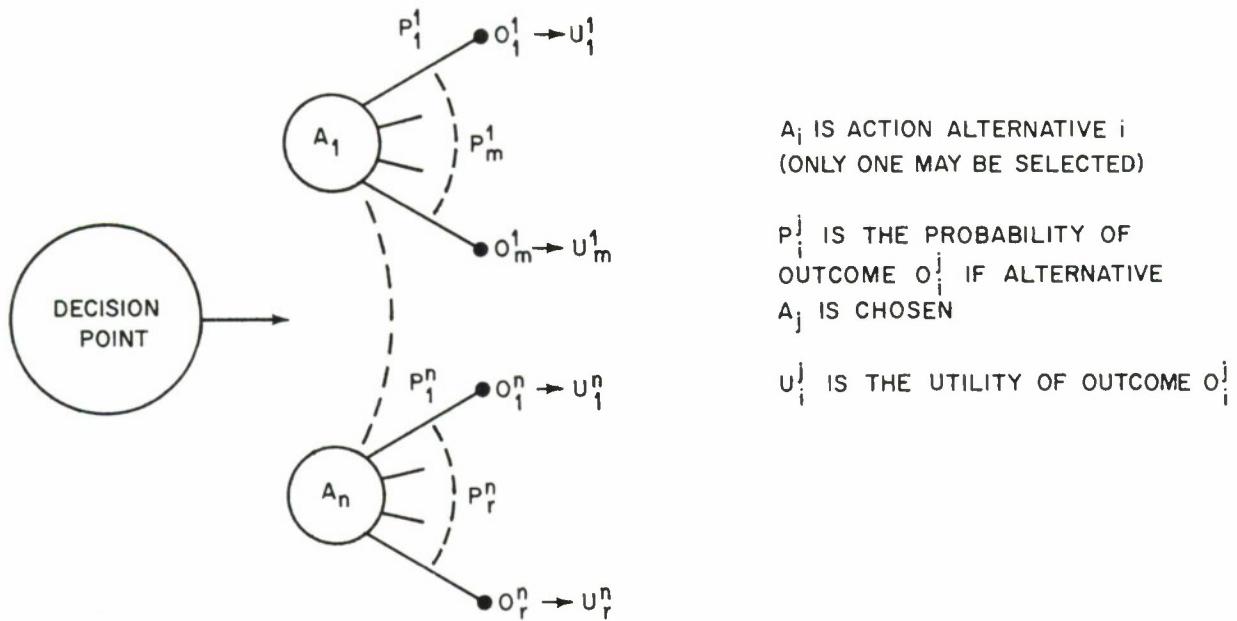
Figure C-5. Structure of a Decision Space

A_i IS ACTION ALTERNATIVE i (ONLY ONE MAY BE SELECTED)

P_i^j IS THE PROBABILITY OF OUTCOME O_i^j IF ALTERNATIVE A_j IS CHOSEN

U_i^j IS THE UTILITY OF OUTCOME O_i^j

A decision space consists of the action alternatives available to the system consistent with its inputs. Exercise designers may structure a decision space so that the system will be confronted by a selected group of action alternatives. Monitoring the alternatives considered by the system does not appear to be an easy task. For instance, if one wants several different aircraft types considered for a particular mission and their availability in specific numbers is indicated to a system by aircraft status reports, one might determine, in debriefing after the exercise, just how many of the available types were actually considered in the course of the exercise. If expectations were not met, one might then try messages stronger than status reports to present alternatives.

Each action alternative has, as its consequence, one or more outcomes, and each of these outcomes has a utility. Utility may be defined as the value to the nation of the particular outcome. Utility is positive in value if it advances national goals and negative if it retards them. The decision maker will often be concerned

with utility values for goals lower than the national, e.g., local command, theatre command, or Air Force command. These utilities may be related to national goals but need not always be consonant with them. (Example: A large raid against a heavily fortified ball-bearing factory may have negative utility value to the bomber wing concerned and to the Air Force if a large loss of aircraft is virtually assured. The utility value of the raid to the nation may, however, be positive because a significant enemy resource would be eliminated. The absolute value of the negative utility of the loss of the aircraft may be less than the positive utility of the enemy's loss of the ball-bearing factory.)

How does the decision maker determine the utility values of the various possible outcomes? They may be indicated to the system by means of guidelines or input messages. Messages may reflect changes in utility which result from outside events or system actions. In the process of selecting one of various action alternatives, one considers not only the utility of possible outcomes of each alternative, but also the probability that each possible outcome will occur if the action is selected.

How are the utility of each possible outcome and its probability of occurrence combined by the decision maker in reaching a rational decision? As an initial suggestion, we propose a formula which has been employed in utility theory for some time. The decision maker will attempt to maximize expected utility $EU(A_j)$ where

$$EU(A_j) = \sum_{1}^{i} P_i^j U_i^j$$

(These terms are defined in Figure C-5, p. C-12.)

Although they are probably implied in the foregoing rule, two analogous rules might be stated. The first states that if one alternative has an outcome whose negative utility is so great that its occurrence would be a catastrophe, that alternative will be eliminated (assuming that the probability of the catastrophic outcome is not negligible). This implies that some limits will have to be imposed on "normal" utility values. The second rule states that if all alternatives contain a

possible catastrophic consequence, then the alternative bearing the lowest probability of a catastrophe is chosen. This formula can be modified by the strategy profiles suggested on p. C-17.

APPLICATIONS

Up to this point we have discussed only the intuitive ideas which would be used in any particular version of the EU model. In this section we shall indicate how these ideas might be applied to provide a model of a particular system and how this model might then be applied to provide the basis for an exercise and evaluation program. The development of such a program can be divided into five sequential steps.

1. <u>Analysis of System</u>. Analyze the system for which the exercise and evaluation program is intended in order to determine (a) the information such a system has and is likely to get in the course of its operations, (b) the kinds of problems such a system is likely to be faced with, and (c) its goals, their interrelations, and the way the system sees its goals. The results of step 1 are a system description which will provide the basis for step 2.

2. <u>Development of Method for Defining Problem Structure</u>. The purpose of step 2 is to produce a reliable technique for going from a particular problem to a description of its probability/utility structure (which in turn provides inputs for the EU model (see Figure C-6)). (The words "problem" and "technique" are defined below.)

A problem for a particular system is, in part, a state of the world in which a situation arises which somehow calls for action by the particular system. Given this part of the problem, one determines what kinds of information the system might conceivably get if the situation did arise. It is this set of information which actually defines the problem for the system and provides the inputs for the technique.

By technique we mean here a manual which can be used by people (of a reasonable level of competence) as a guide to help them produce a description of a given problem in the form of a decision network with relatively precise utilities and
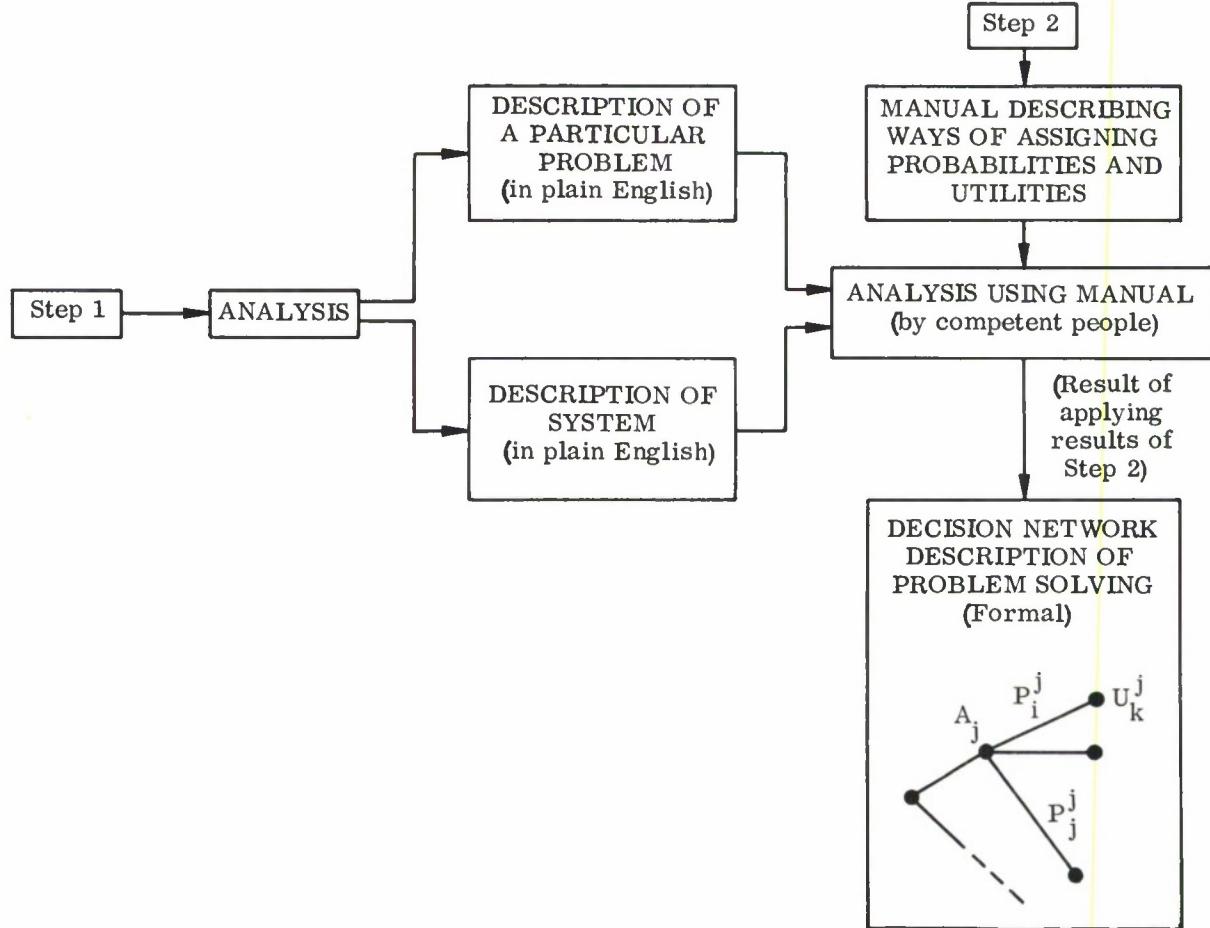
Figure C-6. Roles of Steps 1 and 2

probabilities. The probabilities and utilities here should correspond to what might be called objective probabilities and utilities. That is, they should be the probabilities and utilities one would assign if one had all the information available to the

system, lots of time to think about it, was not influenced by set or emotion, and otherwise used one's best efforts.[*]

3. <u>Validation</u>. Test the validity of the model as a predictor of rational behavior using the manual resulting from step 2. The EU model, aiming to maximize f_1, is supposed to predict the best decisions at decision points. Validating this prediction is done by comparing the results produced by the model (applied to the networks produced by using the manual) with the other criterion for rational behavior, namely the decisions a user makes when he is given the problem from which the network was derived. (The user here can be considered an ideal rational man since it is he who determines what the system should do.)

---

[*]In order to produce such a manual, one keeps in mind the aim for which the resulting structure is intended. In this case, one sees that it is to provide the input for step 3 and acts accordingly. The manual is intended to be used to provide a utility/probability structure which when inserted into the EU model, aiming to maximize

$$\sum_{1}^{n} P_i^j \times U^j = (f_1) \, ,$$

produces the decision that the person assigning the utilities and probabilities would call "sound." For example, the manual might instruct its user to do the equivalent of setting P_i^j to 1 in this formula, henceforth called (f_1), and then try to assign utilities. It might use the following instructions to tell him how to assign utilities to two desirable alternatives A and B: "Assume that you could get either A or B (but not both) simply by asking for them. Call the one you prefer the 'better outcome,' and the other the 'lesser outcome.' Now attempt to determine how many of the 'lesser outcomes' you would require before that number of them made them preferable to one case of the 'better outcome.' Call the number required N. Assign the utility N to the 'better outcome' and 1 to the 'lesser outcome'...."

The method to be used here is as follows (see Figure C-7):
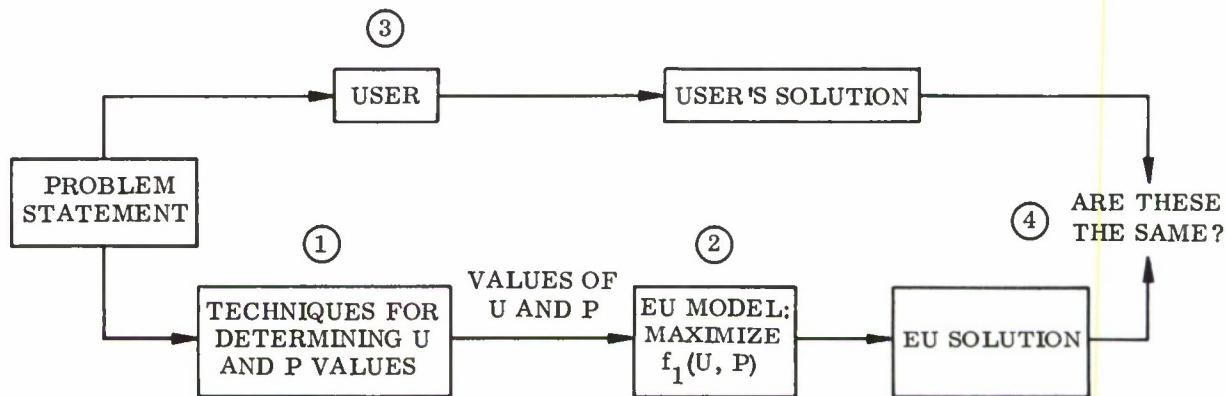
Figure C-7. Validation of EU Model with $f_1(U, P)$ (Step 3)

1. Have a group assign probabilities and utilities to a sample problem using the methods developed as a result of step 2.

2. Insert this structure into the equation f_1 and determine the decisions the EU model predicts.

3. Give the user the problem and ask him to make a decision which he considers "optimal" at each of the choice points in the problem.

4. Compare the results of (3) and (2). They should be approximately the same.

The results of this step are a technique for obtaining, from a description of a problem, a description of an "ideal" behavior. This provides a standard to which actual behavior can be compared.

4. Determination of Strategy Profile.. Run an exercise to produce a system "strategy profile" (see p.C-18 and Figure C-8). In an exercise, a problem (stimulus) is presented to the system and its response to the problem observed. This response maximizes a function (call it f_2) of the utility/probability structure of the problem. This function can be compared to the ideal behavior (maximizing f_1). The outcomes of step 4 are both the divergence between the functions f_1 and f_2

(an answer to the question "How good was the system's performance?") and a guess at the function f_2 itself, which answers the question "What is the structure of the system's behavior?" (What kind of personality does it have?). Each of these outcomes serves as the basis for a different next step (5a and 5b).
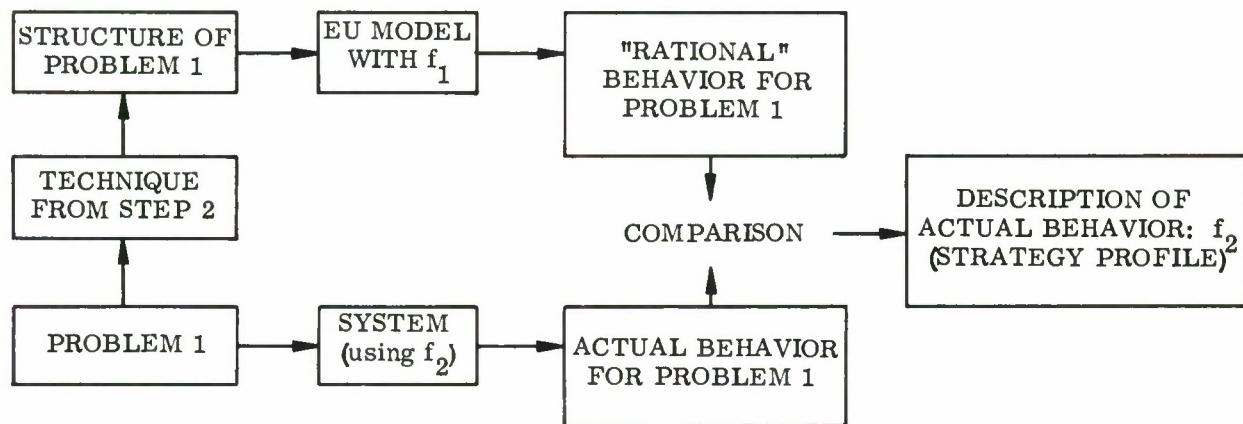
Figure C-8. Producing a Strategy Profile (Step 4)

5a. <u>Validation of Profile</u>. Test the strategy profile of the system (produced as a result of step 4) for validity. This is done experimentally by seeing whether the corrected system strategy (expressed by the desire to maximize f_2 rather than f_1) is consistent from problem to problem. One gives the system a new problem in an exercise, and applying the new function to the probability/utility structure of the new problem one tries to predict the behavior of the system. If this prediction works, this is a step toward the validation of the model with the strategy profile (see Figure C-9).

One problem in validating system strategy profiles is that the system may learn during the repetitions required for validation, and thus the system profile may change during a series of exercises intended to determine the profile. This factor should be considered, but there seems to be no way to avoid its contaminating the validation experiment.

When this step has been successfully performed, one has to some degree validated the extended model (with f_2) as a predictor of system response to
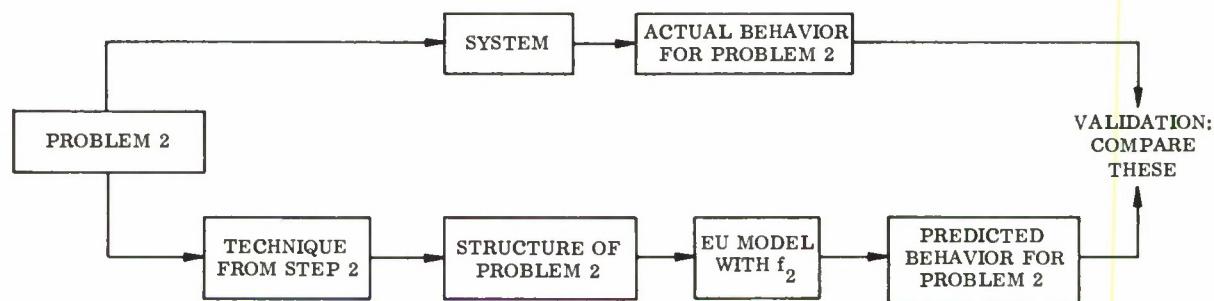
Figure C-9. Validation of EU Model with f_2 (Step 5a)

a large class of problems. These general predictions can be presented to the user, and he can then determine whether this is the kind of behavior he wants. If it is not, one can proceed as in step 5b.

5b. Recommendations. Make recommendations for system changes, either to overcome the divergence between the observed system behavior and the idealized behavior or to meet the desires of the user as expressed as a result of step 5a. Here one can use the categories provided by the model to analyze the source of the unwanted behavior and determine what to do about it. There are at least three alternatives:

a. Exercise: Above we observed that a system's strategy profile may not be constant because the system may learn as it is faced with new problems. In other words, with practice the system's strategy profile may approach the ideal behavior. One may thus choose to exercise the system to bring its behavior up to the desired standards.

b. Change System Inputs: In predicting system behavior, one assumes that the system has the same picture of the utilities and probabilities as those ex- tracted by observers using the problem description and the manual. That is, one assumes that the system's subjective utilities and probabilities (U_s and P_s) are the same as the objective utilities and probabilities (U_o and P_o). But there are many ways in which this may not come about. If the kinds of information which the system

gets during an exercise are those it would get in a similar real situation, then something may be wrong with the ways in which it gets (or displays) this information. That is, there may be some error in the mappings: $P_o \rightarrow P_s$ or $U_o \rightarrow U_s$. There are a number of ways in which these mappings could be changed. Among these are to change message formats, to change the system inputs (sensors, message sources, and the like) and to change system mission statements (which is how the system is told what its goals are).

c. <u>Change System Processing</u>: Even if a system has the proper utilities and probabilities, it may still handle (process) them incorrectly. If this is the case, the processing methods might be changed. One way of changing them is by education of the type discussed under 5a on p. C-18. Other ways include the change of operating procedures and more direct instruction in the proper way to operate.

REFERENCES

1. John Von Neumann and Oskar Morgenstern, "Theory of Games and Economic Behavior" (Princeton, New Jersey: Princeton University Press, 1953).

2. Ward Edwards, "The Prediction of Decisions Among Bets," Journal of Experimental Psychology, <u>50</u>, 201-214 (1955).

3. Ward Edwards, "The Theory of Decision Making," Psychological Bulletin, <u>5</u>, 380-417 (1954), and "The Reliability of Probability Preferences," American Journal of Psychology, <u>67</u>, 68-95 (1954).

4. J. H. Proctor, "Normative Exercising: An Analytical and Evaluative Aid in System Design," paper delivered at the First Congress on the Information System Sciences, Session XV, 20 November 1962, Hot Springs, Virginia.

5. H. Garfinkel, "The Rational Properties of Scientific and Common Sense Activities," Behavioral Science, <u>5</u>, 72-83 (1960).

1. Command and Control Systems
2. Analysis
3. Models, Simulation
4. Flow Diagramming
5. Information Retrieval
6. Decision Making
I. Contract No. AF19(628)-2455, Project No. 2801
II. Technical Operations Research, Burlington, Massachusetts
III. Peter Kugel and Martin F. Owens

(over)

IV. Report ESD-TDR-64-195
V. Aval from OTS
VI. In DDC collection

Directorate of Systems Design, Electronic Systems Division, Air Force Systems Command, L. G. Hanscom Field, United States Air Force, Bedford, Massachusetts. Report ESD-TDR-64-195. SOME TECHNIQUES TO HELP IMPROVE METHODS FOR EXERCISING AND EVALUATING COMMAND AND CONTROL SYSTEMS. Summary report, Jan 64, 115 p. incl. 32 illus., 4 tables.

Unclassified Report

This report describes the development of four techniques intended to improve methods for exercising and evaluating command and control systems: (1) Exercise flow diagram (generalizes and extends the techniques of flow charting for use in exercising), (2) Resource assignment model (uses techniques of

(over)

mathematical logic to describe the information processing in the command and control system), (3) Expected utility model (uses techniques drawn from decision theory to describe decision making in a command and control system), (4) Finite automaton model (uses techniques from the theory of finite automata to describe the relationships between a command and control system and the system exercising it). (Appendixes present three of these models in sufficient detail to permit the reader to apply them. The finite automaton model will be discussed in a separate report.) A program is outlined for developing these models during the second part of the contract period.

Directorate of Systems Design, Electronic Systems Division, Air Force Systems Command, L. G. Hanscom Field, United States Air Force, Bedford, Massachusetts. Report ESD-TDR-64-195. SOME TECHNIQUES TO HELP IMPROVE METHODS FOR EXERCISING AND EVALUATING COMMAND AND CONTROL SYSTEMS. Summary report, Jan 64, 115 p. incl. 32 illus., 4 tables.

Unclassified Report

This report describes the development of four techniques intended to improve methods for exercising and evaluating command and control systems: (1) Exercise flow diagram (generalizes and extends the techniques of flow charting for use in exercising), (2) Resource assignment model (uses techniques of

(over)

1. Command and Control Systems
2. Analysis
3. Models, Simulation
4. Flow Diagramming
5. Information Retrieval
6. Decision Making
I. Contract No. AF19(628)-2455, Project No. 2801
II. Technical Operations Research, Burlington, Massachusetts
III. Peter Kugel and Martin F. Owens

(over)

mathematical logic to describe the information processing in the command and control system), (3) Expected utility model (uses techniques drawn from decision theory to describe decision making in a command and control system), (4) Finite automaton model (uses techniques from the theory of finite automata to describe the relationships between a command and control system and the system exercising it). (Appendixes present three of these models in sufficient detail to permit the reader to apply them. The finite automaton model will be discussed in a separate report.) A program is outlined for developing these models during the second part of the contract period.

IV. Report ESD-TDR-64-195
V. Aval from OTS
VI. In DDC collection